Set Theory for Computer Science

Sandjai Bhulai

Sandjai Bhulai
VU University Amsterdam
Department of Mathematics
De Boelelaan 1111
1081 HV Amsterdam
The Netherlands
s.bhulai@vu.nl

Set Theory for Computer Science

Sandjai Bhulai

Preface

You have finished secondary school and are about to begin at a university of technical college. You want to study computing. The course includes some mathematics – and that was not necessarily your favorite subject. But there is no escape: a certain amount of finite mathematics is a required part of the first-year curriculum, because it is a necessary toolkit for the subject itself.

That is where this book comes in. Its purpose is to provide the basic mathematical language that you need to enter the world of the information and computer sciences. You can learn the essentials here and perhaps have even fun doing so. The book contains certain tools that we need to apply over and over again when thinking about computations. They include:

1. Collecting things together (set theory).
2. Comparing things (relations).
3. Associating one item with another (functions).
4. Recycling outputs as inputs (recursion and induction).

Without an understanding of these basic concepts, large portions of computer science remain behind closed door. As you begin to grasp the ideas and integrate them into your thought, you will also find that their application extends far beyond computing into many other areas.

Sandjai Bhulai
Amsterdam, 2014

Minor modifications have been made by Sander Dahmen in 2024.

Contents

Chapter 1

Sets

Goals of this chapter:

- Build up sets through enumeration, description, union, intersection, and complements.
- Use Venn diagrams to depict formulas with sets or to determine the number of elements.
- Apply fundamental rules of computation for sets.
- Recognize and create a partition of a set.

1.1 What is a set?

Basic concepts, notation
People collect many things: stamps, baseball cards, pictures of mills, etc. The notion of a set is thus a daily concept, however, at the same time it is an essential part of the mathematical language. In this book we see a set as an imaginary collection of objects. These objects are called elements (members) of the set. Synonyms for the notion of a set are collection, family, or class.

A set can be represented by summing up the elements between curly brackets { …}:

Planets := { Mercury, Venus, Earth, Mars, Jupiter, Saturn, Uranus, Neptune }
DaysOfWeek := {Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday}

MainColours := { Red, Green, Blue }
Digits := { 0, 1, 2, 3, ..., 9 }
PrimeNumbers := { 2, 3, 5, 7, 11, 13, ...}

The last two examples are more suggestive in summing up the elements. More often it is easier to describe the elements of a set: the planets of a solar system, the days of the week, the main colours, etc. The notation is then as follows:

Months := { x : x is a month }
MultiplesOfTwo := { x : x is an even natural number }

The general form is

{ x : description of x }, or { x | description of x },

which is to be read as: the sets of all elements x that satisfy the description. The symbol x can be interpreted as the name of a generic element ("prototype").

Naming of sets

Sets usually get a name, such that one can point easily at the collection. In each of the examples, one can find the name on the left. The symbol ':=' is the assignment or definition operator. For instance, the description

DivisorsOf10 := { x : x is a divisor of 10 }

assigns the name 'DivisorsOf10' to the set followed after the symbol ':='. The formula

MonthsOf30:= { April, June, September, November }

assigns the rather natural name to the set of months with 30 days.

The names can be chosen in any way, however, it is preferable to choose them suggestively. Sometimes, neutral symbols such as A, B, C, ...are used for general sets. Some mathematical sets have a standard symbol:

$\mathbb{N}$:= { 0, 1, 2, 3, 4, ...} (the set of all natural numbers)
$\mathbb{Z}$:= { ..., -3, -2, -1, 0, 1, 2, 3, ...} (the set of all integer numbers)
$\mathbb{Q}$:= { x : x is a rational number } (the set of all rational numbers)
$\mathbb{R}$:= { x : x is a real number } (the set of all real numbers)

Rational numbers are also called fractions.

1.2 Elements and subsets

Elements of a set

We have introduced the set Planets in the previous section. One of these

elements is Mars. We can write this as follows:

Mars $\in$ Planets.

The symbol '$\in$' can be read as "is element of", or "is member of". For instance, "Mars $\in$ Planets" can be read as

Mars is an element of the set Planets,
Mars is a member of the set Planets, or
Mars belongs to the set Planets.

The symbol '$\in$' is called the element-of symbol. If the element x belongs to the set A, we write that as $x \in A$. When x does NOT belong to A, we write that as $x \notin A$ (a crossed-out '$\in$'-symbol).

Ganymedes $\notin$ Planets.

Subsets of a set

The set { Mercury, Venus, Earth } consists of element belonging to the set Planets. We say that { Mercury, Venus, Earth } is a subset of the set Planets and write this as

{ Mercury, Venus, Earth } $\subseteq$ Planets.

Here are some other examples. Every digit 0, 1, ..., 9 is a natural number. Hence

Digits $\subseteq \mathbb{N}$.

A month has 28 to 31 days. As a consequence

$\{ x : x \text{ is a month with 30 days} \} \subseteq$ Months.

The symbol '$\subseteq$' is the inclusion symbol. It can be read as "is a subset of" or "is contained in". A set A is a subset of set B when every element of A is also an element of B. We write $A \subseteq B$ when A is a subset of B. Note that $A \subseteq A$ is always true. When A is NOT a subset of B, we write $A \not\subseteq B$ (a crossed-out '$\subseteq'$ symbol). For instance,

{ Sun, Moon, Earth } $\not\subseteq$ Planets.

Equality of sets

Two sets are equal if they have exactly the same elements. For example, the set $\{1, 2, 3, 4, 5\}$ and $\{5, 2, 4, 1, 3, 2, 3\}$ are equal. We write this as

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

Note that repetitions and the order in summing up the elements are irrelevant; they do not contribute to the information of the set, and they sometimes can make the notation difficult to read. The symbol '$=$' is called the equality symbol. Note that the equality symbol is different from the definition symbol

':='. When two sets A and B are equal, we write $A = B$. If two sets are NOT equal, we write $A \neq B$ (a crossed-out '=' symbol). For instance,

MainColours $\neq$ {Red, Yellow, Blue}.

An empty set does not have any element. For example { } and { x : x is a prime number and $24 \leq x \leq 28$ } are empty sets. Since empty sets contain exactly the same elements (namely, none), they are the same. Hence, there is only one empty set en we write that as $\emptyset$. As a side remark: $A \subseteq B$ and $B \subseteq A$ can be true simultaneously; this exactly means that $A = B$.

Number of elements
When a set is given, relevant information is provided by the number of elements belonging to the set. The number of elements of a set A is given by $\#A$ (in some books one uses $|A|$ or $\underline{A}$). For instance,

#Planets = 8,
#DaysOfWeek = 7,
#{ 0, 1, 0, 1, 0, 1 } = 2,
#$\emptyset$ = 0.

Equal sets have an equal number of elements. Sets as $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$, and $\mathbb{R}$ are infinitely large. Their "number" of elements cannot be represented with a natural number.

1.3 Operations: union and intersection

Union of two sets
Take the following two sets of planets:

InnerPlanets := { Mercury, Venus, Earth, Mars }
PlanetsWithMoon := { Earth, Mars, Jupiter, Saturn, Uranus, Neptune }.

We can unite these two sets into one big set of all planets that are an inner planet or are a planet having a moon:

{ Mercury, Venus, Earth, Mars, Jupiter, Saturn, Uranus, Neptune }.

The result of the union is denoted by InnerPlanets $\cup$ PlanetsWithMoon and is exactly equal to the set Planets. Note that the planet Earth belongs to both the set InnerPlanets as the set PlanetsWithMoon. Even though, this element is mentioned only once in the union.

The union of two sets A and B contains elements that belong to set A or set B. The result of the operation "union" is a set that we write as $A \cup B$. The symbol '$\cup$' is the union symbol. The definition of the union is

$A \cup B := \{\ x : x \in A \text{ or } x \in B\ \}$.

We earlier saw that an element of $A \cup B$ may well belong to both sets A and B. In this context, one also speaks of non-exclusive or inclusive membership. We will return to this in Section 1.4.

Intersection of two sets

Take the following sets EUmemberstates, consisting of all member state of the European Union, and the set of NATOmemberstates, consisting of all NATO member states. Every country is abbreviated by three letters of its name; this is the configuration of February 2014:

EUmemberstates := { AUS, BEL, BUL, CRO, CYP, CZE, DEN, EST, FIN, FRA, GER, GRE, HUN, IRE, ITA, LAT, LIT, LUX, MAL, NET, POL, POR, ROM, SLW, SLV, SPA, SWE, UK }

NATOmemberstates := { ALB, BEL, BUL, CAN, CRO, CZE, DEN, EST, FRA, GER, GRE, HUN, ICE, ITA, LAT, LIT, LUX, NET, NOR, POL, POR, ROM, SLW, SLV, SPA, TUR, UK, USA }.

We can now take the intersection of these two sets, which results in the set of all countries that are a member of the European Union and NATO:

{ BEL, BUL, CRO, CZE, DEN, EST, FRA, GER, GRE, HUN, ITA, LAT, LIT, LUX, NET, POL, POR, ROM, SLW, SLV, SPA, UK }

The result of the intersection is written as EUmemberstates $\cap$ NATOmemberstates en consists exactly of 22 elements. The intersection of set A and set B consists of all elements that are member of both set A and B. The result of the operation "intersection" is a set that is written as $A \cap B$. The symbol '$\cap$' is the intersection symbol. The definition of the intersection is given by:

$A \cap B := \{\ x : x \in A \text{ and } x \in B\ \}$.

More than two sets

We have three sets 1CS, 1LI, and 1IMM consisting of all first-year students of the three programmes CS, LI, and IMM, respectively. The union of these three sets consists of all students that belong to at least one of the three sets. The union of three sets A, B, and C is the set of exactly those elements that belong to at least A, B, or C:

$A \cup B \cup C := \{\ x : x \in A \text{ or } x \in B \text{ or } x \in C\ \}$.

Something similar can done with the intersection. Take the following three sets: HighlyIntelligent consisting of people with an IQ $\geq$ 140, Woman con-

sisting of all female persons, and Physics consisting of all students physics. The intersection of these three sets is a set of people that belong to all three sets simultaneously. Hence, the intersection is thus exactly the set of highly intelligent female persons that study physics. The intersection $A \cap B \cap C$ of three sets A, B, and C is the set having elements that belong to each set A, B, and C:

$$A \cap B \cap C := \{ x : x \in A \text{ and } x \in B \text{ and } x \in C \}.$$

All these definitions can be easily extended to four or more sets.

1.4 The universe and the complement

Universal set
For any given problem, there is a specific type (a set) of objects of interest. The set of stellar objects is contextually relevant when talking about stars, planets, moons, and comets. The set of all books is relevant when studying genres such as biographies, fiction, poetry, science, lexicons, etc. A universal set of universe is a prescribed set of elements that is relevant for a specific problem under study.

Complement
Besides the union and intersection operations, we now introduce a third operation. Given a universal set U and a subset A of U, we define the complement of a set A as the set of all elements (of the universe!) that are not member of A. We denote this set as A' (sometimes one sees A^c). For instance: let the universe U consist of all four-letter words and let A be the set of all words (in U!) with at least two vowels. Then, A' is the set of all words (in U!) with at most one vowel.

Difference of two sets
When the complement operation has been defined, one can also define the difference or relative complement $A \setminus B$ of two sets A and B as follows:

$$A \setminus B := A \cap B'.$$

We can also describe this difference directly as the set of elements that are element of A but not of B. This description is also meaningful when no universe is given. Occasionally, one encounters the operation symmetric difference $A \,\Delta\, B$ of two sets A and B, defined by

$$A \,\Delta\, B := (A \cup B) \cap (A \cap B)'.$$

Elements of $A \,\Delta\, B$ are exactly those objects belong to A or B, but not to

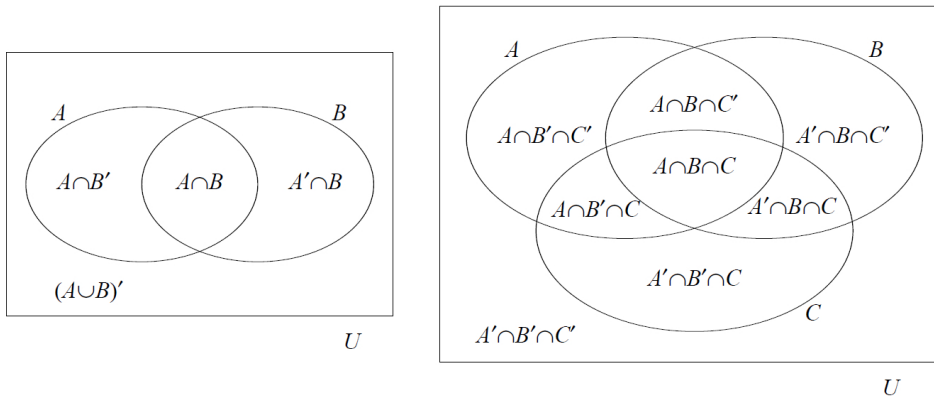


Figure 1.1: Venn diagram with 2 and 3 sets, respectively.

both (exclusive membership). Do not confuse this with the union of two sets. The symmetric difference can also be expressed with the following formula:
$A \Delta B := (A \setminus B) \cup (B \setminus A)$.

Example: let the universe be given by the set of persons. We have two subsets FEW (containing all students at the Faculty of Sciences) and CS (containing all students computer science all over the world). Then the set FEW $\setminus$ CS is the set of all students at the Faculty of Sciences that do not study computer science. Compare it with CS' containing all students (wherever) who do not study computer science.

1.5 Venn diagrams

There is a convenient method to visualize information on different sets simultaneously in a so-called Venn diagram. Such a diagram usually consists of a rectangular areas with several ellipses inside. The different parts in the rectangle represent with different sets.

In the left graph of Figure 1.1 one can see two sets A and B. The left ellipse depicts set A, whereas the right ellipse depicts set B. The total area within the ellipses is $A \cup B$. The common area between the two ellipses is $A \cap B$. The outer area is $(A \cup B)'$. The whole area within the rectangle is the universe U.

Venn diagrams with three or more sets are possible as well (see the right graph in Figure 1.1). The area that corresponds with a formula can be filled

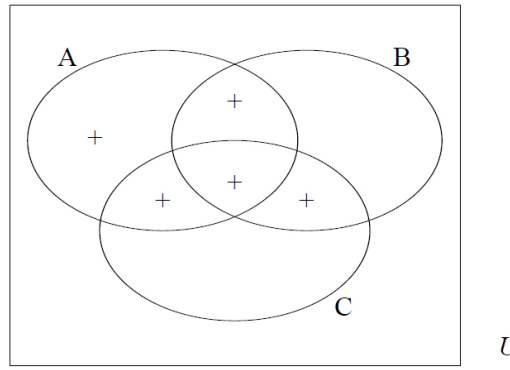


Figure 1.2: Area corresponding to formula $A \cup (B \cap C)$.

in grey. In Figure 1.1 several "atomic" areas have a formula added to them. We additionally give an example with a formula with three sets in a more complicated setting.

Example: The area that corresponds to the formula $A \cup (B \cap C)$ consists of five parts:

$$A \setminus (B \cup C), \qquad (A \cap B) \setminus C, \qquad (A \cap C) \setminus B, \qquad A \cap B \cap C, \qquad (B \cap C) \setminus A$$

The first four parts are areas of A, and the fifth part is the area $B \cap C$ without A. These parts are marked with a '+' symbol in the next Venn diagram (see Figure 1.2).

Application: equalities
A first application of Venn diagrams is to check if two sets described by formulas are equal. Every formula can be represented in a Venn diagram and consequently the areas described can be visually compared. For instance, we have already depicted the set $A \cup (B \cap C)$ in Figure 1.2. Let us now represent the set $(A \cup B) \cap (A \cup C)$, see Figure 1.3. The areas marked with an 'x' depict $A \cup B$, and those with 'y' depict $A \cup C$. Hence, the areas marked with 'x' and 'y' depict $(A \cup B) \cap (A \cup C)$. We can now see that this yields the same area as in the previous figure. Hence, we can conclude that

$$A \cup (B \cap C) = (A \cup B) \cap (A \cup C).$$

Application: Counting of elements A second application of the Venn diagram is related to counting the number of elements. Every part of a Venn diagram corresponds to a formula (e.g., A', $A \cap B$, …). In this part one can also fill in the number of corresponding elements in order to calculate the number of

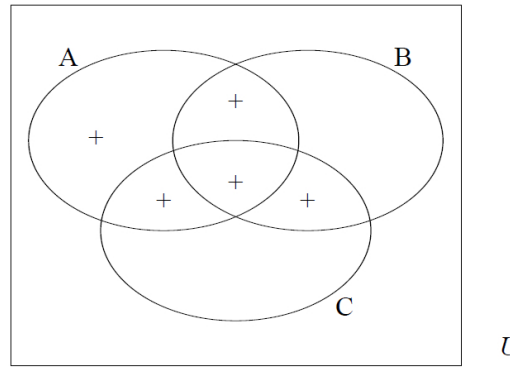


Figure 1.3: Area corresponding to formula $(A \cup B) \cap (A \cup C)$.

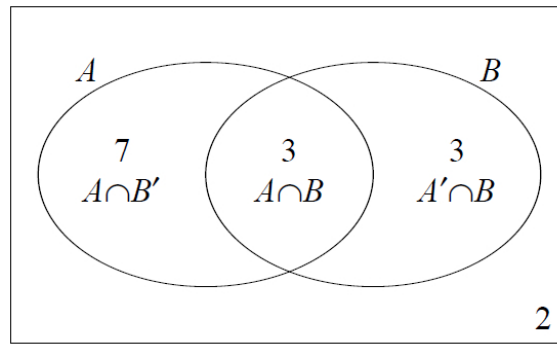


Figure 1.4: Counting the number of elements with a Venn diagram.

elements in the other parts. For instance, take a universe U with 15 elements. There are two subsets A and B, of which $A \cap B$ has 3 elements, A has 10 elements, and B has 6 elements. How many elements does U have outside of $A \cup B$. In Figure 1.4 we have placed the sets as in left graph of Figure 1.1. The number of elements in each part is written in the part itself.

We have first filled in the number of elements in $A \cap B$, which equals 3. Then, we can determine the number of elements in $A \cap B'$, which is 7, and in $B \cap A'$, which is 3. Now, we can see that $A \cup B$ has exactly 13 elements, and that the outer area $(A \cup B)'$ contains 2 additional elements.

1.6 The algebra of sets

In the previous section we demonstrated how Venn diagrams can be used to check the equality of sets described by formulas. In practice this method is only efficient for relatively easy formulas with at most three to four sets. For more complex formulas one can take refuge to algebra of sets.

Algebra
The word "algebra" hints at the presence of operations that satisfy some general principles. We have thus far seen three (main) operations with sets, namely

union	$\cup$	$A \cup B$	(union of A with B)
intersection	$\cap$	$A \cap B$	(intersection of A with B)
complement	$'$	A'	(complement of A)

The operations "difference" and "symmetric difference" are considered secondary operations, since they can be expressed in the operations above. In order to apply the operation complement, a universe $\mathcal{U}$ has been fixed. Several rules for calculation are valid for these operations, such that one can speak of an algebra of sets.

One may compare the situation at hand with a familiar one. Consider the operations $a + b$, $a \cdot b$, and $-a$ with numbers. Arithmetic formulas can often be expressed in more simple formulas using several basic principles and rules, such as $-(a + b) = (-a) + (-b)$ and $(a + b) \cdot c = a \cdot c + b \cdot c$.

Parentheses and priority
The three operations $\cup$ (union), $\cap$ (intersection), and $'$ (complement) can be used to build up more complex formulas in the algebra of sets. In many formulas one needs parentheses '(' and ')' to avoid ambiguity, e.g., when one needs to negate the sum of a and b, one has to write $-(a + b)$ instead of $-a + b$. Too many parentheses in a formula makes it more complex however. Therefore, one agrees on some priority rules to avoid a multitude of parentheses:

- The operation "complement" always has highest priority;
- The operation "union" and "intersection" have equal second priority.

For instance, $A \cup B'$ means $A \cup (B')$, because the complement has priority over union. If one needs the complement of $A \cup B$, one needs to write $(A \cup B)'$.

Laws, the substitution rule

Commutativity:	Idempotence:
$A \cup B = B \cup A$	$A \cup A = A$
$A \cap B = B \cap A$	$A \cap A = A$
Associativity:	Complement:
$A \cup (B \cup C) = (A \cup B) \cup C$	$A \cup A' = U$
$A \cap (B \cap C) = (A \cap B) \cap C$	$A \cap A' = \emptyset$
Distributivity:	De Morgan's laws:
$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$	$(A \cup B)' = A' \cap B'$
$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$	$(A \cap B)' = A' \cup B'$
Identity:	Domination:
$A \cup \emptyset = A$	$A \cup U = U$
$A \cap U = A$	$A \cap \emptyset = \emptyset$
Involution:	
$(A')' = A$	

Substitution rule: to replace some part of a formula with an equal formula.

Table 1.1: Laws of the algebra of sets.

Just as in the algebra of numbers, a small number of principles are needed to calculate set theoretic formulas. The laws are summed up in Table 1.1 and are fundamental to the algebra of sets. They can be easily established with the use of Venn diagrams. Remember, U is the universe and $\emptyset$ is the empty set.

It is important to recognize a law when it occurs in a formula. One often sees a formula with different notation. For example:

$$C \cap (D \cup A) = (C \cap D) \cup (C \cap A)$$

is the second distributive law, in which A, B, and C are replaced with C, D, and A, respectively. It is also possible that at the place of a symbol a complete formula appears. For instance:

$$A' \cup (B \cap C) = (A' \cup B) \cap (A' \cup C)$$

is the distributive law in which the symbol A has been replaced with the formula A'.

The substitution rule
We have added, for sake of completeness, a fairly evident principle to the list of laws: the substitution rule. This rule allows one the replace part of

formulas with equal formulas. An example of this is given by the equality
$(A \cup B) \cap C = (B \cup A) \cap C$

which is valid because the right part of the formula can be obtained by replacing $A \cup B$ with the equal formula $B \cup A$ (commutative law).

Arithmetic with sets
The laws of Table 1.1, together with the substitution rule, describe the so-called algebra of sets. With the data in the table it is, in principle, possible to do any calculation with sets. The following example resembles the rule for arithmetic on the product $(a+b)\cdot(c+d)$ for sets instead of numbers in which the operations add/multiply has been replaced with union/intersection. $(A \cup B) \cap (C \cup D) = ((A \cup B) \cap C) \cup ((A \cup B) \cap D)$
$= ((A \cap C) \cup (B \cap C)) \cup ((A \cap D) \cup (B \cap D))$.
In the second line, the parts $(A \cup B) \cap C$ and $(A \cup B) \cap D$ were replaced with equal expressions according to the distributive laws.

1.7 Partitions

Families of sets
Remember that a set is a collection of imaginary objects. These objects can be anything; we have seen a lot of examples already. We have not seen one particular possibility, however: sets are themselves legitimate objects too, and can consequently be collected as well. Because a "set of sets" does not sound too well, we may alternatively use expressions like

- A collection of sets
- A family of sets
- A class of sets

We can collect all sets that we have introduced in the beginning of this chapter:

{ Planets, DaysOfWeek, MainColours, Digits, PrimeNumbers, Months }.

This is a collection of 6 sets.

Splitting (partitioning) a set
Sets are often being split into smaller subsets. For instance, the set DaysOfWeek can be split in a rather natural manner into the following two subsets.

Weekdays := { Monday, Tuesday, Wednesday, Thursday, Friday }
WeekendDays := { Saturday, Sunday }

First, we may observe that these subsets are disjoint, that is: they do not have any element in common:

Weekdays $\cap$ WeekendDays $= \emptyset$.

More generally formulated: two sets A and B are disjoint when $A \cap B = \emptyset$, i.e., they do not have any elements in common. Second, we also observe that the sets Weekdays and WeekendDays unite to the original set DaysOfWeek:

Weekdays $\cup$ WeekendDays = DaysOfWeek,

Each day of the week belongs to exactly one of the two subsets.

In a more complex situation, we may have several subsets $P_1, P_2, P_3, \ldots$ which can be pairwise disjoint:

$P_1 \cap P_2 = \emptyset, \quad P_1 \cap P_3 = \emptyset, \quad P_2 \cap P_3 = \emptyset, \quad$ etc.

For instance, the subsets

$P_1 := \{2,4,6,8\}, \quad P_2 := \{0,1,9\}, \quad P_3 := \{3,5,7\}$

are pairwise disjoint. We can also conclude that their union is the set Digits:

$P_1 \cup P_2 \cup P_3 =$ Digits.

Each digit belongs to exactly one of the sets P_1, P_2, and P_3.

Partition of a set

The previous observations form the essence of the notion of a "partition". A partition of a set V is a collection of non-empty subsets of V (the "parts" of the partition) such that each element of V belongs to exactly one of the parts. To put it differently, a partition of a set V is a family of non-empty, pairwise disjoint subsets of V, such that the union is equal to V.

Examples: We have a partition

{ Weekdays, WeekendDays }

of the set DaysOfWeek into two parts Weekdays and WeekendDays. Every day belongs to exactly one of these two parts. There is a partition

$\{\{2,4,6,8\},\{0,1,9\},\{3,5,7\}\}$

of the set Digits into three parts. Every digit belongs to exactly one of these three parts.

The summation formula

We take a closer look at the above mentioned partition of Digits. We know that #Digits = 10. This number is now divided over the different parts of the partition:

$$\#\{2,4,6,8\} = 4, \quad \#\{0,1,9\} = 3, \quad \#\{3,5,7\} = 3.$$

With respect to the given partition, we indeed find that $10 = 4 + 3 + 3$. In general, we can say that if the sets $\{P_1, P_2, \ldots, P_n\}$ form a partition of V, then

$$\#V = \#P_1 + \#P_2 + \cdots + \#P_n.$$

This is the summation formula for partitions.

Example: We have a partition of the set Alphabet into two subsets:

Vowels := { 'a', 'e', 'i', 'o', 'u' }
Consonants := { 'b', 'c', 'd', 'f', 'g', 'h', 'j', 'k', 'l', 'm', 'n', 'p', 'q', 'r', 's', 't', 'v', 'w', 'x', 'y', 'z' }

Now, we have #Alphabet = 26, and this is exactly the sum of the parts:

#Vowels + #Consonants = 5 + 21.

1.8 Exercises

1. Write the following sets in the curly-bracket notation by means of summing up the elements or by description. Choose the most convenient of the two.
   (a) The set of all integers that are divisible by 7 or 11.
   (b) The set of all roots of the polynomial $x^3 - 6x^2 + 11x - 6$.
   (c) The set of all solutions $< x, y >$ of the set of equations

$$\begin{cases} 3x - y = 0, \\ x + 2y = 7. \end{cases} \tag{1.1}$$

2. In this exercise we work with the universe $U := \{1, 2, \ldots, 30\}$ with three subsets named MultiplesOf2, MultiplesOf3, and MultiplesOf5 (described suggestively by the names). Give the formula for each of the following sets using the union, intersection, and/or complement using the defined sets.
   (a) Multiples of 2 that are not a multiple of 5.
   (b) Multiples of 6.
   (c) Multiples Odd numbers that are divisible by 3.
   (d) { 30 }.

3. Take the set $\mathbb{R}$ of real numbers as universe and study the following sets

$$A := \{x : 0 < x \leq 2\},$$
$$B := \{x : 1 \leq x < 3\}.$$

Describe the sets $A \cup B$, $A \cap B$, $(A \cup B)'$, and $(A \cap B)'$.

4. Make a Venn diagram for each of the two given formulas. Clearly indicate which area is described by the formulas. Are the two sets equal?
   (a) $(A \cup B) \setminus C$, $(A \setminus C) \cup (B \setminus C)$.
   (b) $(A \,\Delta\, B) \cap C$, $(A \cap C) \,\Delta\, (B \cap C)$.
   (c) $A \cup (B \,\Delta\, C$, $(A \cup B) \,\Delta\, (A \cup C)$.
   (d) $A' \,\Delta\, B'$, $(A \,\Delta\, B)'$.

5. Show using Venn diagrams that
   (a) $(A \,\Delta\, B) \,\Delta\, C = A \,\Delta\, (B \,\Delta\, C)$.
   (b) $(A \,\Delta\, B) \cup (A \,\Delta\, B') = U$.

6. From an infinite number of nickels n (5 cents), dimes d (10 cents), and quarters q (25 cents) three coins are sampled in sequence. A sample such as "nickel, dime, nickel" is represented symbolically as "ndn".
   (a) Describe the universe U of all results and determine $\#U$.
   (b) Describe the subset V of all elements of U that together form a sum of at least 50 cents and determine $\#V$.
   (c) Describe the complement V' of V and determine $\#V'$.

7. A disease Z occurs in 5% of the population, and two symptoms A and B appear to be related to the disease. In a study the following conclusions are drawn. With people that are ill, 45% have symptom A, 70% have symptom B, and 15% has neither of them. With healthy people symptom A appears 10% of the time, symptom B appears 30% of the time, whereas 65% none of the symptoms appear.

   We apply the above information on an arbitrary population of 10.000 people. Depict the information for this group in one Venn diagram, and use this to determine which diagnosis is the best predictor for disease Z: symptom A, symptom B, of the two together.

   Hint: for a given combination (0, 1, or 2) of symptoms, the ratio of #(ill) / #(sample) is meaningful.

8. The law of absorption says

   $A \cap (A \cup B) = A$ (with intersection),
   $A \cup (A \cap B) = A$ (with union).

   These equalities can be checked by using the algebra of sets on the left part of the equality reducing it to the right part.

(a) Check the law of absorption for intersection.
(b) Check the law of absorption for union.

9. Check the following formulas using the algebra of sets. Formulas that contain the symmetric difference 'Δ' need to be converted into union '$\cup$', intersection '$\cap$', and complement first.
(a) $(A \cup B) \cap (C \cup D) = (A \cap C) \cup (A \cap D) \cup (B \cap C) \cup (B \cap D)$.
(b) $(A \cap B) \cup (C \cap D) = (A \cup C) \cap (A \cup D) \cap (B \cup C) \cap (B \cup D)$.
(c) $A \,\Delta\, A = \emptyset$ and $A \,\Delta\, A' = U$.
(d) $(A \,\Delta\, B)' = (A \cap B) \cup (A' \cap B')$.
(e) $A' \,\Delta\, B' = A \,\Delta\, B$.

10. The set NLnumberplates contains all series of symbols in one of the forms.

$$CC\ LL\ LL \quad LL\ CC\ LL \quad LL\ LL\ CC,$$

where an L is an arbitrary capital letter (more specific, a consonant) and a C is an arbitrary digit.
(a) Let P_1 be the subset of NLnumberplates consisting of all digits up front. Define in analogy P_2 (digits in the middle), and P_3 (digits at the end). Check that this is a partition of NLnumberplates.
(b) Use part (a) to calculate #NLnumberplates, a formula suffices.

11. Let W be the set of all three-letter words with at least one vowel. Take for P_1 the subset of all words with a vowel as first letter. Define P_2 in analogy (a vowel a ssecond letter) en P_3 (a vowel as third letter).
(a) Determine $\#W$, $\#P_1$, $\#P_2$, and $\#P_3$.
(b) Is $\{P_1, P_2, P_3\}$ a partition of W?

12. Given are the following sets (note that $0 \in \mathbb{N}$ by definition).

$P_0 := \{3n \mid n \in \mathbb{N}\}$;
$P_1 := \{3n+1 \mid n \in \mathbb{N}\}$;
$P_2 := \{3n+2 \mid n \in \mathbb{N}\}$.

Check that $\{P_0, P_1, P_2\}$ is a partition of the set $\mathbb{N}$ of natural numbers.

1.9 Background

The founder of set theory is Georg Cantor (1845–1918). He developed his ideas concerning sets around 1880. The symbols '$\in$', '$\cup$', '$\cap$' were

introduced by the Italian mathematician Giuseppe Peano (1858–1932). Venn diagrams are named after the British person John Venn (1834–1923), who introduced the method in 1894. The laws on complement of the union and the intersection are named of the British mathematician Augustus DeMorgan (1806–1871).

Cantor's set theory is also termed the naive set theory (zie [3] and [6]). It is indeed a little 'naive': the extensive use of the possibilities to construct sets has led to the paradox of Bertrand Russell (1872–1970) from 1902. In this paradox the fact that sets are objects that can be used as elements in new sets is used in a smart way. By doing so, Russell defines a set R of all objects x for which it holds that $x \notin x$. The answer to the question if $R \in R$ is neither yes nor no.

The contemporary set theory is build up through axioms and avoids all known paradoxes. Most of the axioms are related to the laws in the algebra of sets in Table 1.1. For the purposes of this course, the naive set theory suffices. Reference [9] is explicitly mentioned for computer science students.

Chapter 2

Relations

Goals of this chapter:

- Correctly use finite lists of data, length of lists, and position of elements.
- Construct and use the Cartesian product of two or more sets.
- Construct and use binary or multi-set relations (data banks).
- Perform arithmetic with binary relations: compositions and inverses.
- Recognize important properties of binary relations (reflexivity, symmetry, anti-symmetry, transitivity).

2.1 Lists, position, and length

Basic concepts, notation

A list is an enumeration of objects in a certain sequence. These objects are the elements (members) of the list. Repetitions in lists are allowed. Lists are denoted by enumerating the elements between angle brackets $< \ldots >$ in the correct sequence:

PlanetList := <Mercury, Venus, Earth, Mars, Jupiter, Saturn, Uranus, Neptune>

DaysOfWeekList := <Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday>

JustAList := <'l', 'e', 't', 't', 'e', 'r', 'w', 'o', 'o', 'r', 'd'>

DigitList := <0, 1, 2, 3, ..., 9>

NumberList := <0, 1, 2, ...>

Note that the quotes around the letters (such as in 'a', 'b', ...) are commonly used (also in programming languages) in order not to confuse with one-letter variables. Every thinkable object can be member of a list. A list of lists is also a valid construction. For instance, a word is (essentially) a list of letters, and one can construct a list of words. Also the empty list $<>$ is allowed. The lists in the above mentioned examples have been assigned to a name (just as with sets) using the definition symbol ':='.

Position and length

Every listed object has a position in the list, denoted by the sequence number of the element in the enumeration. For example, in the list of planets, the planet Earth has the third position and Uranus the seventh. The total number of positions is called the length of the list. The list with the days of the week has length 7, en the empty list has length 0. Lists that are infinite in length are also possible, e.g., the list NumberList of all natural numbers in sequence of increasing numbers.

A list of length two is called a pair or a tuple, a list of length three is called a triple. In general: a list of length n is called an n-pair. When the lists is not too length, the elements are called the coordinates of the list. The n-th coordinate is the element at position n. For example, in the pair $< a, b >$ the element a is the first coordinate, and b the second coordinate.

Example: a date consists of at least a number for the day and one for the month, usually mentioned in this order. Think of, e.g., "2 October' denoted as 02-10. Mathematically seen, this is a pair $< 02, 10 >$. Hence, '10 October' is represented as $< 10, 10 >$ (remember that repetition is allowed in lists!). One can also have an extensive date, such as 'Thursday 2 October'. This is a triple $<$ Thu, 02, 10 $>$. Even more specific is 'Thursday 2 October 2004'. This is a 4-pair $<$ Thu, 02, 10, 2004 $>$. In the same manner: time is usually denoted by a list with hours and minutes. Often one also mentions the day, week, etc. The 5-pair $<$ Mon, 09, 45, 11, 30 $>$ is a representation of the time 'Monday between 9:45am and 11:30am'.

Equality of lists

Two lists L_1 and L_2 are equal, symbolically $L_1 = L_2$, if they have the same objects at the same position. Hence, the same number of objects are enumerated in the same sequence. Consequently, equal lists have the same length. Thus, we can see that

$< 1, 1 > \neq < 1 > \quad < 02, 10 > \neq < 10, 02 >$.

In the first case the length of the two lists are different, in the second list the

order in which the elements are mentioned is different ('2 October' is not the same as '10 February').

2.2 Cartesian product

Recall that a set is a collection of objects. Lists are legitimate objects to be "collected". Think of the set of all lists of digits, or the set of all words, i.e., non-empty, finite lists of letters. We want to collect lists in a more structured manner.

Example: a date such as "27 January" (or "27-01") can be seen a pair (a list of length two) $< 27, 01 >$ with its first coordinate the number of the day, and as second coordinate the number of the month. Hence, let us define two sets:

DayNumbers := { 01, 02, 03, ..., 31 },
MonthNumbers := { 01, 02, 03, ..., 12 }.

We can create the set Data as the set of all pairs (lists of length two) with its first coordinate an element of DayNumbers and its second coordinate an element of MonthNumbers. A suggestive notation for this set of dates is:

DayNumbers $\times$ MonthNumbers.

Product of two sets
The set Data is an example of a Cartesian product of two sets, DayNumbers and MonthNumbers. In general: a Cartesian product of two sets A and B is defined as the set of alle pairs $< a, b >$ with $a \in A$ and $b \in B$, and is denoted by $A \times B$. Hence:

$A \times B := \{< a, b > : a \in A \text{ and } b \in B\}$.

The set A is the first factor and the set B is the second factor.

In Figure 2.1 the Cartesian product of $\{0, 12\}$ and $\{a, b, c\}$ is depicted. The pairs are shown at the right location in the graph. The first coordinate can be read from the horizontal axis, and the second coordinate from the vertical axis.

Product of more than two sets
The date can be extended with a third number, the year. For this purpose, introduce the set YearNumbers, and consider a date to be a triple with first and second position as defined as before, and the third position an element of the set YearNumbers. The set of all data that is established in this way, is the Cartesian product of three sets:

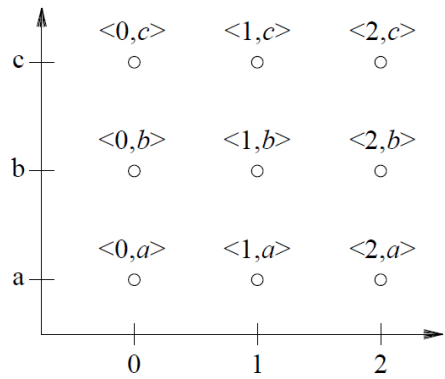


Figure 2.1: Cartesian product $\{0,1,2\} \times \{a,b,c\}$.

DayNumbers × MonthNumbers × YearNumbers.

A typical element of this set is the triple $< 16,07,2004 >$.

In general, one has n sets $A_1, A_2, \ldots, A_n$, and one constructs the set of all lists $< a_1, a_2, \ldots, a_n >$ of length n with element a_1 from the set A_1, element a_2 from the set A_2, etc. The resulting Cartesian product is denoted by $A_1 \times \cdots \times A_n$. This gives:

$$A_1 \times A_2 \times \cdots \times A_n := \{< a_1, a_2, \ldots, a_n > : a_1 \in A_1, a_2 \in A_2, \ldots, a_n \in A_n\}.$$

In this product A_i is the i-factor $(i = 1, \ldots, n)$. When one takes n times the Cartesian product of the same set A, then one writes the Cartesian product as A^n (A to the power n). This is the exponential notation.

Example: we have the sets DaysOfWeek, Hours, and Quarters := { 00, 15, 30, 45 }. An interval in a schedule can be seen as a 5-pair of the form

<day, start_hour, start_quarter, end_hour, end_quarter>.

Hence, one could have a lecture in the schedule as $< \text{Tue}, 9, 00, 10, 45 >$. The set of such 5-pairs is the Cartesian product

DaysOfWeek × Hours × Quarters × Hours × Quarters.

The exponential notation can be found in the following examples:

$\mathbb{R}^2 := \mathbb{R} \times \mathbb{R}$ (the plane); a typical element is $< 3, -2 >$,
$\mathbb{R}^3 := \mathbb{R} \times \mathbb{R} \times \mathbb{R}$ (the space); a typical element is $< 3, -2, 5 >$.
Alphabet^4 (all 4-letter words); a typical element is <'w', 'o', 'r', 'd'>.

The product formula

Suppose that two finite sets A and B are given. It is easily seen that there are $(\#A) \cdot (\#B)$ ways to construct a pair $< a, b >$ with $a \in A$ and $b \in B$. As

a consequence:

$\#(A \times B) = \#A \cdot \#B$.

This product formula also is valid for a Cartesian product with more than two factors:

$\#(A_1 \times A_2 \times \cdots \times A_n) = \#A_1 \cdot \#A_2 \cdot \cdots \cdot \#A_n$.

In case all sets $A_1, A_2, \ldots, A_n$ are equal to the same set A, the product formula yields $\#(A^n) = (\#A)^n$. Hence, one can see that the exponential notation for sets (A^n) corresponds to the classical powers for numbers $((\#A)^n)$.

Examples: There are 31 numbers for the days, and there are 12 numbers for the month. How many pairs of the type

<daynumber, monthnumber>

can one construct? According to the product formula, the answer is $31 \cdot 12 = 372$. Depending on whether we have a leap year or not, we have six or seven dates that do not make sense in the Cartesian product

Data := DayNumbers × MonthNumbers.

The set Alphabet of all small letters a to z has 26 elements. The set of all four-letter words is Alphabet^4, and has according to the product formula $26^4 = 456,976$ elements.

2.3 Relations with n positions

Relations as sets

In practice, one almost daily encounters relations between a variety of subjects: people, objects, time, places, …. For example, personal results of exams relate certain subjects and certain grades, and can be represented as pairs of the form

<subject, grade>.

A schedule for college consists of courses with corresponding teachers, time schedules, number of lecture halls, en weeknumbers; more precisily, it consists of 5-pairs of the form:

<week, time, teacher, course, room>,

and thus relates five objects of different types.

A relation sometimes is decribed in words (e.g., relations in family between brothers, aunts, …) or with formulas (e.g., relations between numbers such as $x^2 + y^2 + z^2 = 1$). Sometimes a relation is described by enumeration of the elements. This is the case with the results of exams and with the

schedule of colleges.

Relations can thus be seen as sets. In the following definition $n \geq 2$ represents a natural number. A relation with n factors is a set with n-pairs (recall that these are lists of length n). The number n is also called the arity of the relation R. if R is a subset of an n-ary Cartesian product $A_1 \times \cdots \times A_n$ of sets $A_1, \ldots, A_n$, then R is a relation with n factors and the product set $A_1 \times \cdots \times A_n$ is called the type of the relation. In this case, we speak of a typed relation.

For an n-ary relation R, the formula $< a_1, \ldots, a_n > \in R$ is also read as the n-pair $< a_1, \ldots, a_n >$ that is in relation R.

Examples: The personal results of the exam form a 2-ary (binary) relation of type

Courses $\times$ Grades,

and the schedule of lectures is a 5-ary relation of type

Weeks $\times$ Times $\times$ Teachers $\times$ Courses $\times$ Rooms.

The relation between numbers given by $x^2 + y^2 + z^2 = 1$ is a 3-ary relation of type $\mathbb{R}^3$.

Description of a relation
When seen as a set, a relation can also be defined by a description. This is the case, e.g., for the relation

$\{< x, y, z > : x^2 + y^2 + z^2 = 1\}$.

The description "$x^2 + y^2 + z^2 = 1$" is also called the description of the relation. A relation R with n factors can therefore be represented as:

$\{< x_1, \ldots, x_m > \ :$ description of the relation for $x_1, \ldots, x_n \}$.

Example: a date is a triple with as first coordinate an element from DayNumbers, second coordinate an element from MonthNumbers, and the third coordinate an element from YearNumbers. There is a relation RealData consisting of all triples that are legitimate dates according to the calendar. The description of the relation is rather complicated with a lot of restrictions on the number of the days in each month also depending on the leap year.

Databases
Relations of a given type also appear as "relational databases". In their simplest form, these are tables with several columns, one for each factor that appears in the relation type. The schedule of the lectures are an example of

a database with five columns (a 5-ary relation). In column 1 we have the list of weeks, in column 2 a time (day, start, and end). In column 3 the teacher is listed, in column 4 the course, and in column 5 the room.

2.4 Binary relations

In this section we focus our attention to simple relations that are used a lot in practice, namely, relations with arity 2. Such relations are also called binary relations. When a binary relation R is assigned to a type, then this is necessarily of the form $A \times B$, where A and B are sets. In this situation, we have $R \subseteq A \times B$, and we can say that R is a relation between elements of A en of B (in this order).

Example: the next relation describes which muses correspond to which domains of art according to the old-Greek mythology.

{<Clio, History>, <Euterpe, Song>, <Euterpe, Elegiac poetry>,
<Thalia, Comedy>, <Melpomene, Tragedy>, <Terpsichore, Dance>,
<Erato, Lyric Poetry>, <Polyhymnia, Hymns>, <Urania, Astronomy>,
<Calliope, Epic poetry> }

This relation consists integrally of pairs and is therefore binary. This lengthy enumeration can be better represented in a table.

Muse	Domain
Clio	History
Euterpe	Song, Elegiac poetry
Thalia	Comedy
Melpomene	Tragedy
Terpsichore	Dance
Erato	Lyric Poetry
Polyhymnia	Hymns
Urania	Astronomy
Calliope	Epic poetry

Different pairs can sometimes be joined on one line (see, e.g., the two domains of Euterpe). Relations within a family are often binary relations:

IsBrotherOf := $\{ < x, y > : x$ is a brother of $y \}$.

After the introduction of the set People, one can define the relation IsBrotherOf to be of type People $\times$ People. Another example is

FollowsCourse := $\{ < x, y > : x \in$ Students and $y \in$ Courses and x follows $y \}$.

The relation FollowsCourse is of type Students $\times$ Courses.

Infix notation
With binary relations we use the so-called infix notation:

$x\,R\,y$ instead of $< x, y > \in R$.

In combination with a good choice for the name, the notation for the relation is suggestive:

x IsBrotherOf y instead of $< x, y > \in$ IsBrotherOf
x FollowsCourse y instead of $< x, y > \in$ FollowsCourse

The infix notation is very common in mathematical relations such as IsEqualTo (symbol $=$) and IsSmallerThanOrEqualTo (symbol $\leq$). An additional advantage is that the information on series of pairs can be represented more efficiently: compare the following:

$5^2 \leq 3 \cdot 9 \leq 85/3 \leq 30$,

with the more tedious

$< 5^2, 3 \cdot 9 > \in$ IsSmallerThanOrEqualTo,
$< 3 \cdot 9, 85/3 > \in$ IsSmallerThanOrEqualTo,
$< 85/3, 30 > \in$ IsSmallerThanOrEqualTo.

2.5 Different representations of binary relations

In order to work effectively with relations, a good representation is important. For instance, in Calculus one has been taught to depict real-valued functions with graphs. Mathematical concepts such as maximum, minimum, zero, extreme points get a more visual and geometric meaning. In this section we discuss three representations that are useful for finite binary relations.

First representation: Venn diagrams
Suppose that a binary relation $R \subseteq A \times B$ is given for finite sets A and B. Thus: R is of type $A \times B$. The sets A and B can be represented by the usual ellipse, but apart from each other. Elements of A and B are depicted by a dot in the corresponding ellipse. Now, an arrow is drawn from dot $a \in A$ to a dot $b \in B$ when a and b are related in R, i.e., $< a, b > \in R$.

The relation, depicted in Figure 2.2, can be summarized through enumeration by

$\{< p_1, q_1 >, < p_1, q_5 >, < p_3, q_1 >, < p_3, q_3 >, < p_4, q_4 >, < p_5, q_6 >,$
$< p_6, q_5 >\}$.

and it is one of type $A \times B$ with $A := \{p_1, p_2, \ldots, p_6\}$ and $B := \{q_1, q_2, \ldots, q_6\}$.

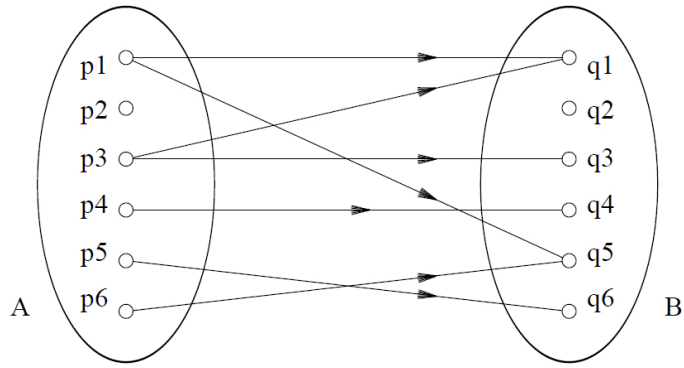


Figure 2.2: Representation with a Venn diagram.

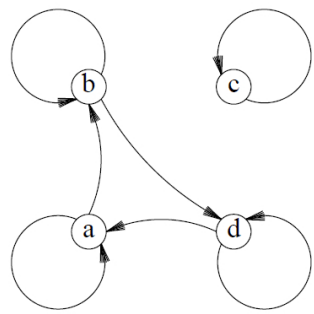


Figure 2.3: A directed graph with four vertices.

Second representation: directed graphs
A directed graph (or a digraph) is a set of vertices connected with edges. The vertices are usually represented by circles. A relation of type $A \times B$ with A and B finite can be represented by a directed graph by drawing a vertex for every element of $A \cup B$. An edge from vertex $a \in A$ to vertex $b \in B$ denotes that $< a, b >$ are in relation. This representation is used a lot when the types A and B are not explicitly given, or when $A = B$.

The relation represented in Figure 2.3 is

$$\{< a,a >, < b,b >, < c,c >, < d,d >, < a,b >, < b,d >, < d,a >\}$$

and is a relation of type $\{a, b, c, d\}^2$.

Third representation: 0/1 matrices and relation tables
Suppose that a relation R of type $A \times B$ is given with A and B finite sets,

i.e., $R \subseteq A \times B$. We provide an order to the elements of A and B (remember that a set does not have an order for enumeration). Now, we can create a relation table, of which the rows are tied to the elements of A in the specified order, and of which the column are tied to the elements of B in the specified order. We place a '1' at the entry for row $a \in A$ and column $b \in B$ when $a\,R\,b$ and a '0' otherwise. Alternatives are to put 'T' (of 'true') instead of '1', and 'F' (of 'false') instead of '0'. Another alternative is to place '+' instead of '1', and to place '-' instead of '0'. The matrix representation of R follows from deletion of the names for the rows and the columns. This matrix is also called the relation matrix of R.

Example: Take the following two sets:

$A := \{a, b, c, d\}$ $B := \{1, 2, 3, 4\}$,

with the provided order of enumeration. In the schemes that follow, we first give the relation table of the binary relation R of type $A \times B$ en next we provide the matrix representation (the relation matrix). of R by deletion of the names for the rows and columns. By a 0/1 matrix one understands a rectangular scheme of zeros and ones.

Relation table:

R	1	2	3	4
a	0	1	0	1
b	0	0	0	0
c	1	1	1	1
d	0	0	1	1

Matrix representation:

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

In this example one can see that $a\,R\,2$ is valid, but not $a\,R\,1$.

2.6 Some constructions with binary relations

Inverse relation

Consider the relation $R :=$ IsParentOf between people. With the infix notation $x\,R\,y$ we denote that x is parent (father or mother) of y, in correct notation $< x, y > \in$ IsParentOf. Now, a pair is a list, and thus sensitive for order: x first, and then y. It is thus a different relation than S which relates y to x. This relation has as description:

$y\,R\,x$ holds exactly when x IsParentOf y.

Which relation S is this? Common sense gives the answer:

y IsChildOf x holds exactly when x IsParentOf y.

We call the relation IsChildOf the inverse of the relation IsParentOf.

In general, the inverse relation S of a relation R is constructed by reversing all pairs in relation R: $x\,R\,y$ gives $y\,R\,x$, and vice verse. The 'neutral' notation

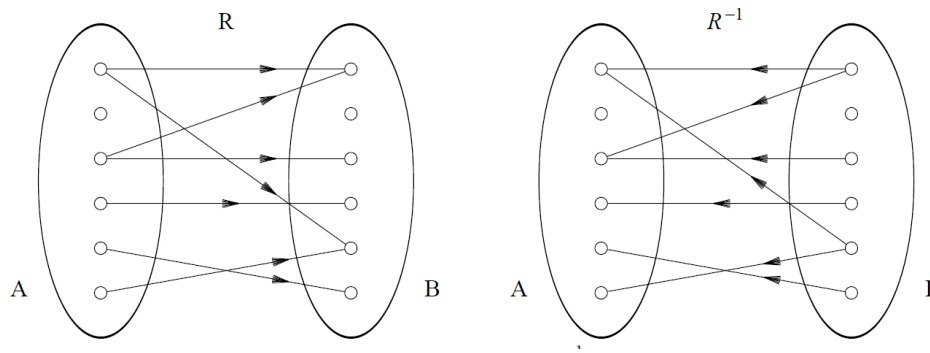


Figure 2.4: Inverse relation R^{-1} of R.

R^{-1} (read: the inverse of R) is common for the inverse relation of R. Hence:
$R^{-1} := \{< x, y >:< y, x >\in R\}$.
For instance, the inverse relation of IsParentOf is the relation IsChildOf:
$(\text{IsParentOf})^{-1} = \text{IsChildOf}$.

Example: Consider the relation LivesIn of type Persons *times* Places. When Jan lives in Amsterdam, this can be represented in the following two ways.

Jan LivesIn Amsterdam <Jan, Amsterdam>$\in$ LivesIn.

Reversely, one could say that Amsterdam is the residence place of Jan. The relation that appears can be called IsResidenceOf:

Amsterdam IsResidenceOf Jan <Amsterdam, Jan>$\in$ ResidenceOf.

Conclusion: the inverse relation of LivesIn is the relation IsResidenceOf, en we can represent this by

$(\text{LivesIn})^{-1} = \text{IsResidenceOf}$.

The type of the inverse relation IsResidenceOf is Places $\times$ Persons. Recall that, in comparison with the type of the relation LivesIn, the factors are in reversed order. This is a general phenomenon: if R is of type $A \times B$, then R^{-1} is of type $B \times A$. As a side note: it is a fairly easy observation that the inverse of the inverse yields the original relation. Thus: $(R^{-1})^{-1} = R$.

Representation with Venn diagrams
If the relation R is given by means of a directed graph or by means of a Venn diagram, then the inverse relation R^{-1} is given by reversing the arrows in the graph (see Figure 2.4).

Representation with matrices

If the relation R is given in matrix form, then the inverse relation $R{-}1$ is given by transposing the matrix, i.e., change the rows into columns and vice verse, with the same order. For example: take the two sets

$$A := \{a, b, c, d\}, \quad B := \{1, 2, 3, 4\},$$

with the given order of enumeration. Furthermore, we define the relation R by

$$\{< a, 2 >, < a, 4 >, < c, 1 >, < c, 2 >, < c, 3 >, < c, 4 >, < d, 3 >, < d, 4 >\}$$

of type $A \times B$ from the example of the third representation. The matrices of R and R^{-1} are given by

$$R : \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}, \qquad R^{-1} : \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}.$$

In the left matrix, the rows correspond to a, b, c, d (in this order). In the right matrix, the rows corresponds to $1, 2, 3, 4$ (in this order). The rows of the left matrix have become the columns of the right matrix and vice verse.

Composite relation

Expressions such as "the uncle of a friend" and "the grandson of a neighbour of an aunt" are not unusual in conversation. These are, in fact, a series of relation or composite relations. Let us study this in more detail.

Example: Focus on the relation of all pairs $< x, y >$ with description "x is an uncle of a friend of y". Assume that Aad is uncle uncle of Bea, and that Bea is a friend of Cor. We can represent that in two ways:

Aad IsUncleOf Bea <Aad, Bea>$\in$ IsUncleOf,
Bea IsFriendOf Cor <Bea, Cor>$\in$ IsFriendOf.

We now make a composition. Aad is an uncle of a friend of Cor.

the pairs <Aad, Bea> and <Bea, Cor> are chained to <Aad, Cor>.

A general description of this phenomenon is as follows. Let R and S be given relations. The composition of R after S exists exactly of those pairs $< x, z >$ that result when there is an 'intermediate' element y such that the pair $< x, y >$ in S can be 'chained' to the pair $< y, z >$ in R:

the pairs $< x, y >$ and $< y, z >$ are chained to $< x, z >$.

The notation for this composite relation is $R \circ S$, to be read as R after S. The operator used for this has the symbol '$\circ$', and the operator is called the composition operator. The definition for the composite relation is

$R \circ S := \{< x, z >:$ there is a y with $x\, S\, y$ and $y\, R\, z\}$.

First an S-pair, then an R-pair chained to each other. Hence, the reason to read R after S.

The example of IsFriendOf as R and IsUncleOf as S yields
IsFriendOf $\circ$ IsUncleOf :=
$\{< x, z >:$ there is a y with x IsUncleOf y and y IsFriendOf z $\}$.

Hence, we find that <Aad, Cor>$\in$ IsFriendOf $\circ$ IsUncleOf. The intermediate element is Bea. There could be more than one intermediate elements: maybe Cor has two friends that are nieces of each other, with Aad as common uncle. Please do note that difference in description and notation:

- An uncle of a friend
- IsFriendOf after IsUncleOf
- IsFriendOf $\circ$ IsUncleOf

The order of the components when using $\circ$ is quite essential.

Here are some other examples:

IsMarriedTo $\circ$ IsParentOf = IsParentInLawOf
IsParentOf $\circ$ IsMarriedTo = IsParentOf.

Note that the incorrect use of the composition can lead to absurd relations. For instance, the composition of 'the residence of a teacher' is meaningful, but the composition 'the teacher of a residence' is the empty relation.

Representation with Venn diagrams
If R is a relation of type $B \times C$, and S is a relation of type $A \times B$, then the situation for a composite relation $R \circ S$ can be depicted well with Venn diagrams. A pair $< a, c >$ with $a \in A$ and $c \in C$ are in relation $R \circ S$ exactly then if there is a path of arrows starting in a leading to c via an element of $b \in B$. In Figure 2.5 there are exactly 8 pairs in the composite relation.

As a side note. When the relations R and S are given by matrices, then the composite relation $R \circ S$ is in essence given by the matrix product S with R (denoted by $S \cdot R$). This is not exactly a relation matrix, since there could be entries that are greater than 1. These entries need to be replaced by a 1.

Multi-composite relations
The result of a composition is a new binary relation, and therefore the process of composition can be repeated. A composition turns out to be associative: for the composition of three relation R, S, and T

$(R \circ S) \circ T = R \circ (S \circ T)$.

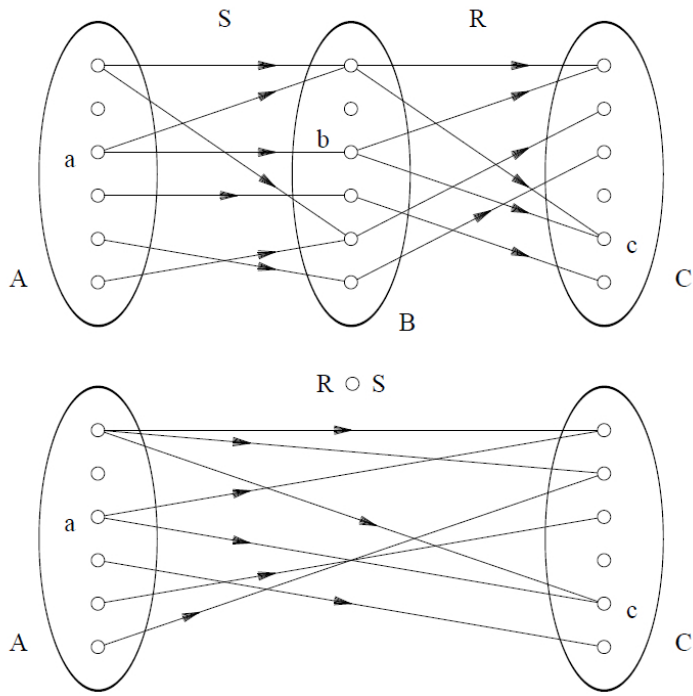


Figure 2.5: Composite relation $R \circ S$ with $a\,S\,b\,R\,c$.

Take as example the description "the grandson of a neighbour of an aunt". It is given that

Aart IsGrandsonOf Betty <Aart, Betty>∈ IsGrandsonOf
Betty IsNeighbourOf Carla <Betty, Carla>∈ IsNeighbourOf
Carla IsAuntOf David <Carla, David>∈ IsAuntOf

When chained together, we get: Aart is grandson of a neighbour of an aunt of David. Hence, <Aart, David>∈ IsAuntOf ∘ IsNeighbourOf ∘ IsGrandsonOf. The intermediate elements are Betty and Carla. This multi-composite relation can be build up in two steps, and in two manners. First, Betty is a neighbour of an aunt of David (the intermediate element is Carla):

<Aart, David>∈ (IsAuntOf ∘ IsNeighbourOf) ∘ IsGrandsonOf

with intermediate element Betty. Second, Aart is grandson of an aunt of Carla (the intermediate element is Betty):

<Aart, David>∈ IsAuntOf ∘ (IsNeighbourOf ∘ IsGrandsonOf)

with intermediate element Carla. We can see that the composition is asso-

ciative:

$$(\text{IsAuntOf} \circ \text{IsNeighbourOf}) \circ \text{IsGrandsonOf} = \text{IsAuntOf} \circ (\text{IsNeighbourOf} \circ \text{IsGrandsonOf}).$$

Inverse of composite relations
The composition of two binary relations is a binary relation. One can therefore study the inverse of this relation. There is a simple rule: the inverse of the composite relation is the composite in reverse order of the inverse relations:

$$(R \circ S)^{-1} = S^{-1} \circ R^{-1}.$$

Example: Take the relation with the description "a residence of the teacher of". Given is that $< a, b > \in$ IsResidenceOf and $< b, c > \in$ IsTeacherOf. This means that a is a residence of b, who is teacher of c, and consequently, a is a residence of a teacher of c; in formal notation

$$< a, c > \in \text{IsTeacherOf} \circ \text{IsResidenceOf}.$$

The inverse relations can be named as follows:

$$\text{LivesIn} = (\text{IsResidenceOf})^{-1}$$
$$\text{IsStudentOf} = (\text{IsTeacherOf})^{-1}$$

The inverse of the composite relation now expresses that c is a student of someone that lives in a:

$$< c, a > \in \text{LivesIn} \circ \text{IsStudentOf} = (\text{IsResidenceOf})^{-1} \circ (\text{IsTeacherOf})^{-1} = (\text{IsTeacherOf} \circ \text{IsResidenceOf})^{-1}.$$

Indeed: c is a student of b, and b lives in a.

2.7 Relations in a set

Relations in a family, such as 'brother' and 'uncle' are binary relations of the type People $\times$ People and the relation IsSmallerThan is of the type $\mathbb{R} \times \mathbb{R}$. In all these cases the type of the relation is the square of the set. There is a special name for this phenomenon. If V is a set, then a relation of the type $V \times V$ is called a relation in V.

Examples: The relation IsBrotherOf is a relation in the set of People. The relation IsSmallerThan is a relation in the set $\mathbb{R}$. The relation LivesIn is of type People $\times$ Places, and can therefore not be seen as a relation in a set.

In this section, we are interested in the following properties of a relation R in a set V.

Reflexivity: every element of V is with itself in relation R.

Symmetry: if an element of V is in relation R with a second element of V, then the second element is in relation R with the first element.

Anti-symmetry: if an element of V is in relation R with a second element of V, and the second element is in relation R with the first element, then the two elements are the same.

Transitivity: if an element of V is in relation R with a second element of V, and the second element is in relation R with a third element of V, then the first element is in relation R with the third element.

A convenient reformulation of these descriptions is given in the language of logic. Note that it can happen that one element of V is in relation R with itself, but a different element of V is not. The relation R is then not reflexive. Similar statements can be made for the other three properties as well.

The table hereafter summarizes the properties together with an intuitive example of a relation that satisfies the property. In every example, we consider a relation in a set of all people.

Property	Example
Reflexivity	Knows
Symmetry	IsFriendOf
Anti-symmetry	IsParentOf
Transitivity	IsDecendentOf

- The relation Knows is reflexive, since for every person x it holds that x Knows x.
- The relation IsFriendOf is symmetric: if x IsFriendOf y, then also y IsFriendOf x.
- The relation IsParentOf is anti-symmetric: the case where x IsParentOf y and y IsParentOf x as well does not occur.
- The relation IsDecendentOf is transitive: if x IsDecendentOf y and y IsDecendentOf z, then also x IsDecendentOf z.

2.8 Exercises

1. Product of sets.

   (a) For a specific game one has to throw three dice. The first one has the usual number on each face, the second has a different colour on each face (say, black, wit, red, yellow, green, and blue), and the third has a sign '♡' on three faces and a '♣' on the others. Describe the set of possible throws in this game as a Cartesian product. How many different outcomes are there?

   (b) Given are the sets

   $$A := \{b, c, f\};\ B = \{a, b, c, e, f\};\ C = \{a, c, d\};\ D = \{a, c, e\}.$$

   How many elements are there in the set $(B \times C) \cap (A \times D)$, and how many elements are there in the set $(B \times C) \cap (A \times D)'$?

   (c) Check with the outcomes of part (b) that

   $$\#((B \times C) \cap (A \times D)) + \#((B \times C) \cap (A \times D)') = \#(B \times C).$$

2. Given is a set Throws := Number × Number, with Number := {1, 2, ..., 6}, together with a collection of eleven subsets $P_2, \ldots, P_{12}$. The set with number k (for $k = 2, \ldots, 12$) is defined as

   $$P_k := \{< o_1, o_2 > : o_1, o_2 \in \text{Number}, o_1 + o_2 = k\}.$$

   For example, P_3 is the set with all throws having a sum of 3. Do the subsets for a partition of the set V? Respond with 'yes' or 'no'; however, please explain when the answer is 'no'.

3. Given are two relations R and S in the set $\{a, b, c, d\}$ depicted by the following graphs.

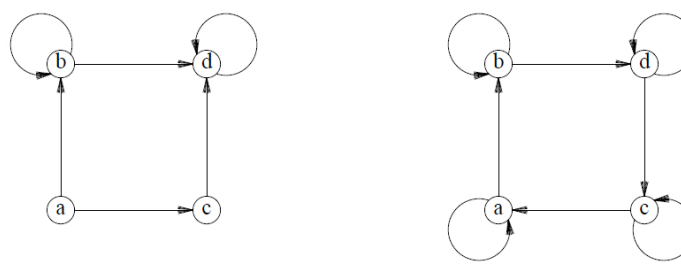


Figure 2.6: Relation R (left) and relation S (right).

(a) Give a description by enumeration of the relations R and S. Also give a representation by 0/1 matrices of both relations.

(b) Draw the directed graphs corresponding to $R \circ S$ and $S \circ R$. Systematically determine if the relations are reflexive, symmetric, anti-symmetric, en/or transitive.

(c) Is $R \circ S = S \circ R$?

4. R and S are two relations in a set with 4 elements in enumerated order. Determine the matrix of $R \circ S$ when R and S are described as follows in matrix form:

$$R: \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \qquad S: \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

Hint: Number the elements in the give order from 1 to 4. Take two numbers $i, j \in \{1,2,3,4\}$ and investigate if $i\,(R \circ S)\,j$ as follows. Fix row number i of S and column number j of R, and check if in one of more positions there is a common '1'.

5. In a set of people are given the relations IsParentOf and IsIdenticalWith (the relation with exclusively pairs $< x, x >$ with x a person). In the following parts one is asked to construct new relations, based on the two relations and the operations composition, inverse, union, intersection, complement of relations.

(a) The relation IsGreatGrandChildOf

(b) The relation IsSiblingOf

(c) The relation IsCousinOf

6. In this exercise we consider a date as an element of the Cartesian product

DayNumbers × MonthNumbers × YearNumbers,

of which the factors are defined as follows.

DayNumbers := {01, 02, ..., 31},
MonthNumbers := {01, 02, ..., 12},
YearNumbers := $\{x : 1990 \leq x\}$.

The set Data consists of all real data according to the calendar. Consider a binary relation Period that consists of all pairs of dates $< d_1, d_2 >$ from Data with $d_1 \leq d_2$ (i.e., date d_1 appears before date d_2 or is equal to it) and the difference $d_2 - d_1$ is at most 21 days.

(a) Which data pairs $< x, y >$ satisfy $< x, y > \in$ Period?

$<< 08, 10, 1998 >, < 20, 09, 1998 >>$;
$<< 10, 10, 1998 >, < 01, 11, 1998 >>$;
$<< 10, 11, 1998 >, < 01, 12, 1998 >>$.

(b) Is the relation Period reflexive? Symmetric? Anti-symmetric? Transitive?

(c) What does the relation Period $\circ$ Period mean?

7. A library registers every loan and the period for the loan. This administration is seen as a 5-ary relation Administration of type

Members $\times$ Books $\times$ Out $\times$ Back $\times$ Data,

of which Members is the set of all library members, Books is the collection of all books in the library, Out := Data, and Back := Data (see previous exercise for a definition of Data). It is the intention that a 5-pair such as

<Jan, NiceBook, $< 15, 09, 1998 >$, $< 04, 10, 1998 >$, $< 24, 09, 1998 >$ >

represents a certain case, namely, that Jan borrowed the book NiceBook on 15-09-98, and has not delivered it yet on 24-09-98, and that Jan is allowed to keep the book until 04-10-98. One can imagine that every day the administration is processed with the mutations of the previous day and that old files are never thrown away.

(a) Suppose that $< x_1, x_2, x_3, x_4, x_5 > \in$ Administration. What can one say about x_3, x_4, x_5?

(b) Give a precise description, with the use of the relation Administration, of the following (intuitive) relations.

i. The relation Borrowed of type Books $\times$ Data.

ii. The relation Available of type Books $\times$ Data.

iii. The relation InViolation of typeMembers $\times$ Books $\times$ Data.

(c) Discuss the possibility of two 5-pairs that are element of Administration but only differ in one position.

2.9 Background

The name 'Cartesian' refers to the French philosopher and mathematician René Descartes (1596–1650), who introduced the coordinate system that allowed to compare points in the plane to two real numbers $< x, y >$. This reduced the 'difficult' geometry of the plane to the 'easy calculus of real numbers.

The rather sober definition of a binary relation as a set of pairs is due to the logician A. DeMorgan (British, 1806–1871) and C.S. Peirce (American, 1839–1914). In 1864 the operations for the inverse and the composition of two binary relations were described by DeMorgan. And in 1870 the logic of binary relations was sketched by Peirce. Around 1940, the famous Polish logician Alfred Tarski (1902–1983) developed the general relation algebra that could describe, apart from the operations union, intersection, and complement, also the composition and inverse of binary relations.

Relation algebra is commonly used in semantics of (imperative) programming languages. The idea behind this is that a computer instruction in a given state (memory value) is transformed into a new state (a different value in memory), and thus creates a binary relation. Consecutive instructions lead to composite relations. The first three chapters from [11] are a representation of the contents of this chapter in the language of relation algebra. Reference [9] is more at the level of this book.

Relations of higher arity are rare in mathematics, but are relevant for the study of relational databases (E.F. Codd, 1970). The columns in the relation table are called attributes in this context. There are special languages to perform operations on relations of high arity. A well-known language is SQL (Structured Query Language), developed in the seventies (see [5], [7]). This language is used in modern database programs.

Chapter 3

Relations: Partial Order

Goals of this chapter:

- Recognize a partial order or a total ordering and their properties.
- Display relations for orderings by Hasse diagram, and vice verse, read information on relation for orderings from a Hasse diagram.
- Reproduce the most frequently used examples for relations for orderings (ordering for numbers, inclusion of sets, Cartesian and lexicographic products).
- Create simple proofs on partial orders by a strategy and a chain of valid arguments.
- Find maxima and minima in subsets of a partially ordered set.

3.1 Refresher: binary relations

Basic concepts

In Chapter 2 we introduced the concept of a relation as a set of n-pairs for a specific integer $n \geq 1$. The number n is called the arity of the relation. Relations are often a subset of a Cartesian product of the form

$$V_1 \times V_2 \times \cdots \times V_n.$$

This product set is called the type of the relation. An element of a relation of the type $V_1 \times V_2 \times \cdots \times V_n$ is thus an n-pair (list of length n, n-pair)

$$< a_1, a_2, \ldots, a_n >,$$

with $a_1 \in V_1, a_2 \in V_2, \ldots, a_n \in V_n$.

We then restricted ourselves to binary (2-pairs) relations. Such relations consist of pairs and are of type $V_1 \times V_2$. Binary relations can be represented by means of directed graphs or by means of matrices. Next, we further restricted ourselves to relations in a set V, i.e., relations of the type $R \subseteq V \times V$.

Relations are sets. The notation $< x, y > \in R$ for a binary relation R is thus meaningful and expresses that x and y are in relation R. We denote this as $x\,R\,y$ (infix notation). Relations can (as with sets) be described by enumeration or by description. For instance, the so-called circle relation is the relation in the set $\mathbb{R}$ of real numbers, and is given by

$$R := \{< x, y > : x^2 + y^2 = 1\}.$$

(This relation is called circle relation because it exactly consists of points $< x, y >$ on the unit circle in the plane $\mathbb{R}^2$. The phrase $x^2 + y^2 = 1$ is the description of the relation R. It is common to write the relation as

$$xRy :\leftrightarrow x^2 + y^2 = 1.$$

The symbol ':↔' (colon, double implications) refers to an equivalence relation. There is some analogy with the symbol for assignment ':=', which refers to a defined equality.

Possible properties of a relation in a set
For relations in a set we have introduced four concepts in Chapter 2

Reflexivity, Symmetry, Anti-symmetry, Transitivity. We formulate the properties here once again, only this time in logic form. We start from a relation R of type $V \times V$.

Property	Formal definition	Example
Reflexivity	$\forall x \in V, x\,R\,x$	IsEquallyOldAs
Symmetry	$\forall x, y \in V, (x\,R\,y \rightarrow y\,R\,x)$	IsFriendOf
Anti-symmetry	$\forall x, y \in V, (x\,R\,y \wedge y\,R\,x \rightarrow x = y)$	IsParentOf
Transitivity	$\forall x, y, z \in V, (x\,R\,y \wedge y\,R\,z \rightarrow x\,R\,z)$	IsDecendentOf

In this and the next chapter we shall focus on a particular set of three of the four properties.

reflexivity + anti-symmetry + transitivity (this chapter)
reflexivity + symmetry + transitivity (next chapter)

3.2 Relations for orderings

Definition
A relation R in a set V is called a partial order when R satisfies the following three properties:

reflexivity, anti-symmetry, and transitivity.

The collection of the set V together with the partial order R is called a partially ordered set. The specification "partial" is often omitted.

Example: Consider the natural ordering in the alphabet. We previously defined the 26 letters of the alphabet as the following list:

<a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z>.

One can see this as a description for a partial order between the 26 elements:

$x \leq y \leftrightarrow x$ is before y in the enumeration or x is equal to y.

Example: Consider the natural ordering of numbers. In the set $\mathbb{N}$ of natural numbers the natural ordering $\leq$ is a partial order. We list the three properties that are required for this relation.

(i) (reflexivity) $\forall x \in \mathbb{N}, x \leq x$.
(ii) (anti-symmetry) $\forall x, y \in \mathbb{N}, (x \leq y \wedge y \leq x \rightarrow x = y)$.
(iii) (transitivity) $\forall x, y, z \in \mathbb{N}, (x \leq y \wedge y \leq z \rightarrow z \leq z)$.

Instead of the set $\mathbb{N}$, one can also focus on the set $\mathbb{Z}$ of integers and on $\leq$ for the natural ordering for integers. This is also a partial order. The natural ordering in the set $\mathbb{Q}$ of rational numbers and the natural ordering in the set $\mathbb{R}$ of real numbers are partial orders as well.

Example: Consider the relation IsDivisorOf between positive numbers

Let P be the set of all integers > 0:
$P := \{z : z \in \mathbb{Z} \text{ and } z > 0\} = \mathbb{N} \setminus \{0\}$.

The relation IsDivisorOf (with symbol '|') in P is described as follows:

$x|y :\leftrightarrow \exists z \in P, x \cdot z = y$.

The relation '|' is a partial order in P.

Example: Consider the power set and the ordering through inclusions. Remember (see Chapter 1) that a set is a collection of arbitrary objects. Therefore, a set itself is a legitimate object and can be collected in a 'larger' collection. For instance, a partition of a set V is a set of subsets of V (with some additional properties). When V is a set, then there exists a set of all subsets of V:

$\mathcal{P}(V) := \{X : X \subset V\}$.

This set is called the power set of V. When X and Y are subsets of V (i.e., when $X, Y \in \mathcal{P}(V)$), then $X \subseteq Y$ means that X is a subset of Y. Then, it is not difficult to recognize the infix notation in this for a binary relation with the symbol $\subseteq$. This leads us to this example: inclusion is a relation in $\mathcal{P}(V)$, and, more specifically, it is a partial order. We list the three properties for this relation:

(i) (reflexivity) $\forall X \in \mathcal{P}(V), X \subseteq X$.
(ii) (anti-symmetry) $\forall X, Y \in \mathcal{P}(V), (X \subseteq Y \wedge Y \subseteq X \rightarrow X = Y)$.
(iii) (transitivity) $\forall X, Y, Z \in \mathcal{P}(V), (X \subseteq Y \wedge Y \subseteq Z \rightarrow X \subseteq Z)$.

Notation and terminology

Relations for orderings are a common phenomenon in daily life: alphabetic orders, small and large numbers, etc. Relations for orderings are used quite frequently in mathematics and in theoretical computer science. Traditionally, the smaller than/equal to symbol '$\leq$' or the symbol '$\subseteq$' is used to describe a partial order. The sentence '$x \leq y$' can be phrased in different ways. We list some common expressions:

'x is smaller than or equal to y'	'y is greater than or equal to x'
'x comes before y'	'y comes after x'

Graphical representation with Hasse diagrams

A partial order defined on a finite set can, as with any binary relation, be depicted by a directed graph. In case of a partial order is it possible to draw the graph such that the order can be read 'from below to top'. For example, in Figure 3.1 we see that

$a \leq d, \qquad b \leq d, \qquad d \leq f.$

In order to depict the information as clearly as possible, the reflexive part of the relation is not drawn, and one includes sufficient arrows for the relations to reconstruct the transitive relations. For example, through the arrows

$a \rightarrow d \rightarrow f$

we can reconstruct the (omitted) arrow $a \rightarrow f$. This 'thinned' representation is called the Hasse diagram of a partial order. The Hasse diagram thus has the minimum information to completely reconstruct the partial order.

In order to construct this diagram, one needs to 'thin' the information to the strict minimum. Here is a precise recipe:

For every point x:
1. Let $Gx := \{y : x < y\}$.

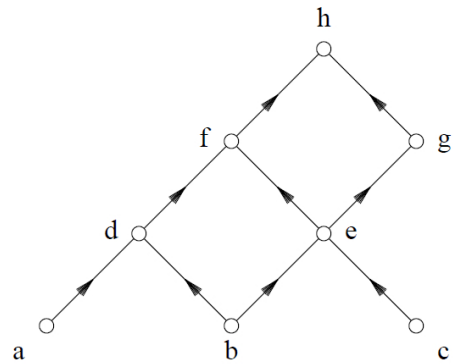


Figure 3.1: A Hasse diagram.

2. For every $y \in Gx$: let $Gy := \{z : y < z\}$ and $Gx := Gx \setminus Gy$.
3. For every $y \in Gx$: draw an arrow between $x \to y$ in the Hasse diagram.

Additional info: for a given $y \in Gx$ and $z \in Gy$, we have $x < y < z$. The arrow $x \to z$ can be omitted, because the information can be recovered from $x \to y \to z$.

Comparability and total ordering

When a and b are two elements in a partially ordered set for which $a \leq b \vee b \leq a$, then the two elements a and b are comparable in the given order. We already saw that in the natural ordering in each of the systems of numbers $\mathbb{N}, \mathbb{Z}, \mathbb{Q}$, and $\mathbb{R}$ is a partial order. In fact, every system of numbers has the property that two numbers are comparable. In the following definition we give this property a name.

A partial order in a set V is called a total ordering (or a linear ordering) on V when

$$\forall x \in V, \forall y \in V, (x \leq y \vee y \leq x).$$

A partial order is thus total when all elements are comparable to each other. The natural ordering in a standard system of numbers is a total ordering. The Hasse diagram of a total ordering is a sequence of arrows. Hence, the name 'linear ordering' that is used as well.

The order, drawn in the Hasse diagram in Figure 3.1 is not total, because the elements a and b are not comparable. For a different example, consider the power set of $V := \{0, 1\}$. This set consists of four elements

$$\emptyset, \{0\}, \{1\}, \{0, 1\}.$$

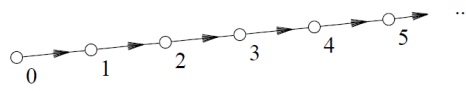


Figure 3.2: Hasse diagram of $\mathbb{N}$ with the natural ordering.

The following holds: $\{0\} \not\subseteq \{1\}$ and $\{1\} \not\subseteq \{0\}$, i.e., the elements $\{0\}$ and $\{1\}$ are not comparable for inclusion. More generally, one can conclude that the ordering by inclusion of $\mathcal{P}(V)$ is not a total ordering as soon as V has two or more elements.

Strict ordering
A partial order is reflexive: equality is included. With every partial order $\leq$ comes a strict partial order $<$, defined by

$$x < y :\leftrightarrow x \leq y \wedge x \neq y.$$

The relation $<$ that one gets is also called the small-than relation. A strict ordering comes forth out of a partial order by omitting the reflexive part. When reversed, a partial order can be constructed by adding the reflexive part to a strict ordering:

$$x \leq y \leftrightarrow x < y \vee x = y.$$

The most frequent ways in which the sentence $x < y$ are phrased are:

'x is (strictly) smaller than y'	'y is (strictly) greater than x'
'x comes strictly before y'	'y comes strictly after x'

A relation R in a set V is called anti-reflexive if no element of V is in relation with itself. Formally: $\forall x, \neg x\, R\, x$. We prove the following:

Theorem 3.1. Given is a partial order $\leq$ in a set V and the corresponding strict ordering $<$. Then, the relation $<$ is anti-reflexive, anti-symmetric, and transitive.

Proof. Checking the anti-reflexivity is left as an exercise for the reader.

We prove that the relation $<$ is anti-symmetric. The goal is to show that for all x, y in V we have: $x < y \wedge y < x \rightarrow x = y$. Let $x, y \in V$ arbitrary, and assume that $x < y$ and $y < x$. Then, we have that $x \leq y$ and $y \leq x$. Hence, $x = y$ based on anti-symmetry of $\leq$.

We now show that the relation $<$ is transitive. Take $x, y, z \in V$ arbitrary, and assume that $x < y$ and $y < z$. We first check that $x \leq z$. From $x < y$ we have that $x \leq y$. From $y < z$ we have that $y \leq z$. Now, from $x \leq y$

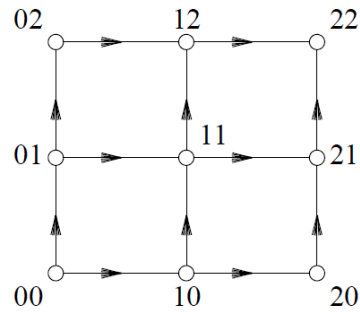


Figure 3.3: Hasse diagram of $\{0,1,2\}^2$ with Cartesian ordering.

and $y \leq z$ we have that $x \leq z$ by transitivity of $\leq$. Next, we check that $x \neq z$. This is done by a proof by contradiction. For this purpose, assume the contrary $x = z$. It then follows that $x < y$ and $y < x$ (since, $y < z = x$. However, from the anti-symmetry of $<$ (see first part of the proof) it follows that $x = y$. However, at the same time it follows from $x < y$ that $x \neq y$. Hence, we have a contradiction. □

3.3 Two constructions of relations for orderings

First construction: Cartesian product
Given are two sets V_1 and V_2, each with their own partial order $\leq_1$ and $\leq_2$, respectively. The Cartesian ordering in the product set $V_1 \times V_2$ is defined by

(CART): $< x_1, x_2 > \leq < y_1, y_2 > \quad :\leftrightarrow \quad x_1 \leq_1 y_1 \wedge x_2 \leq_2 y_2$.

Loosely spoken: in the Cartesian ordering a pair is smaller than or equal to a second pair when the inequalities corresponding to both first and both second coordinates hold.

Example: Let $V_1 := V_2 := \{0,1,2\}$ with the natural (total) ordering where $0 < 1 < 2$. In the Cartesian ordering on $\{0,1,2\}^2$ we have

$< 0,0 > \leq < 0,1 >, < 0,0 > \leq < 1,0 >, < 1,0 > \leq < 1,1 >, < 0,1 > \leq < 1,1 >,$

but not $< 0,1 > \leq < 1,0 >$ and not $< 1,0 > \leq < 0,1 >$. The constructed Cartesian order is not total. In Figure 3.3 the pairs are depicted in short by omitted the angle brackets and commas. For instance, '01' corresponds to the pair $< 0,1 >$.

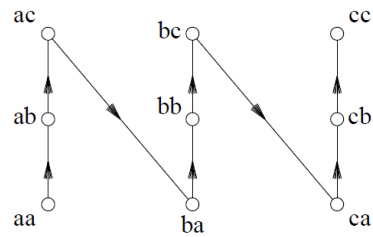


Figure 3.4: Hasse diagram of $\{a,b,c\}^2$ with lexicographic ordering.

Theorem 3.2. Given are two sets V_1 and V_2, each with their partial order $\leq_1$ and $\leq_2$, respectively. The relation (CART) yields a partial order on $V_1 \times V_2$.

Proof. Let $\leq$ be the relation in $V := V_1 \times V_2$ described by (CART). One needs to show that $\leq$ is reflexive, anti-symmetric, and transitive. We only show transitivity of $\leq$.

Let $x, y, z, \in V$ be arbitrary and assume that $x \leq y$ and $y \leq z$. Then, there exists $x_1, y_1, z_1 \in V_1$ and $x_2, y_2, z_2 \in V_2$ such that

$$x =< x_1, x_2 >, \quad y =< y_1, y_2 >, \quad z =< z_1, z_2 >.$$

From $x \leq y$ it follows from the definition of (CART) that (1) $x_1 \leq_1 y_1$ and (2) $x_2 \leq_2 y_2$. From $y \leq z$ it follows from the definition of (CART) that (3) $y_1 \leq_2 z_1$ and (4) $y_2 \leq_2 z_2$. From (1) and (3) it follows (by transitivity of $\leq_1$) that $x_1 \leq_1 z_1$. Similarly, from (2) and (4) it follows (from transitivity of $\leq_2$) that $x_2 \leq_2 z_2$. These conclusions together with the description of (CART) result in $x =< x_1, y_1 > \leq z =< z_1, z_2 >$. □

The construction of the Cartesian ordering can be extended relatively easy to products with more than two ordered sets. With three factors the relation is defined as:

(CART3): $< x_1, x_2, x_3 > \leq < y_1, y_2, y_3 > \quad :\leftrightarrow \quad x_1 \leq_1 y_1 \wedge x_2 \leq_2 y_2 \wedge x_3 \leq_3 y_3$.

Note that Theorem 3.2 remains unchanged.

Second construction: Lexicographic ordering Given are two sets V_1 and V_2, each with their own partial order $\leq_1$ and $\leq_2$, respectively. The lexicographic ordering in the product set $V_1 \times V_2$ is defined by

(LEX): $< x_1, x_2 > \leq < y_1, y_2 > \quad :\leftrightarrow \quad x_1 <_1 y_1 \vee (x_1 = y_1 \wedge x_2 \leq_2 y_2)$.

Loosely spoken: in the lexicographic ordering a pair is smaller than or equal to a second pair when the inequality holds for the first coordinate, and the second coordinate otherwise. Summarized: the first coordinate decides.

Let $V_1 := V_2 := \{'a', 'b', 'c'\}$ with the usual (total) ordering. Then, $V_1 \times V_2$ is the set of all two-letter words with the letters $'a', 'b', 'c'$. In the lexicographic ordering on these words, we find that

$$'aa' \leq 'ab' \leq 'ba' \leq 'bb'.$$

The constructed lexicographic ordering is clearly total. In Figure 3.4 all pairs are displayed in a shortened manner without the parentheses and commas. For instance, 'ab' represents $< a, b >$.

Theorem 3.3. Given are two sets V_1 and V_2, each with their partial order $\leq_1$ and $\leq_2$, respectively. The relation (LEX) yields a partial order on $V_1 \times V_2$. This ordering is total if both the orderings $\leq_1$ and $\leq_2$ are total.

Proof. Let $\leq$ be the relation in $V := V_1 \times V_2$ described by (LEX). The strict orderings corresponding with $\leq_1$ and $\leq_2$ are $<_1$ and $<_2$, respectively. One needs to show that $\leq$ is reflexive, anti-symmetric, transitive, and total. We only show transitivity of $\leq$.

Let $x, y, z \in V$ be arbitrary and assume that $x \leq y$ and $y \leq z$. Then, there exists $x_1, y_1, z_1 \in V_1$ and $x_2, y_2, z_2 \in V_2$ such that

$$x =< x_1, y_1 >, \quad y =< y_1, y_2 >, \quad z =< z_1, z_2 >.$$

It follows from $x \leq y$ and the definition (LEX) that $x_1 <_1 y_1$, or $x_1 = y_1$ and $x_2 \leq_2 y_2$. From $y \leq z$ it follows using (LEX) that $y_1 <_1 z_1$ or $y_1 = z_1$ and $y_2 \leq_2 z_2$. The combination of these two facts leads to four alternatives, which we will check:

(i) $x_1 <_1 y_1$ and $y_1 \leq_1 z_1$: transitivity of $\leq_1$ yields $x_1 \leq_1 z_1$, and thus $< x_1, x_2 >\leq< z_1, z_2 >$ according to (LEX).

(ii) $x_1 <_1 y_1$ and $y_1 = z_1$ and $x_2 \leq_2 y_2$. The strict inequality and equality yield $x_1 <_1 z_1$ and thus $< x_1, x_2 >\leq< z_1, z_2 >$ according to (LEX).

(iii) $x_1 = y_1$ and $x_2 \leq_2 y_2$ and $y_1 <_1 z_1$: The strict inequality and equality yield $x_1 \leq_1 z_1$ and thus $< x_1, x_2 >\leq< z_1, z_2 >$ according to (LEX).

(iv) $x_1 = y_1$ and $x_2 \leq_2 y_2$ and $y_1 = z_1$ and $y_2 \leq_2 z_2$: The equalities yield $x_1 = z_1$. The inequalities and transitivity of $\leq_2$ yield $x_2 \leq z_2$. Hence, $< x_1, x_2 >\leq< z_1, z_2 >$ according to (LEX).

In all cases we find $x =< x_1, x_2 >\leq z =< z_1, z_2 >$. □

The construction of the lexicographic ordering can also be extended to products of two or more sets. The description of this relation is rather cumbersome, but the principle is clear: the symbol at the left decides, in case of equality the next symbol decides, etc. Theorem 3.3 remains unchanged.

In case of a product with n factors, that are all equal to the standard set Alphabet with the usual ordering, one gets the (total) dictionary-ordering for words of length n. The name 'lexicographic ordering' actually refers to this example. Time stamps and dates can also be described through a product set of two or more factor and a lexicographic ordering.

3.4 Maxima and minima

Definitions
Let $(V, \leq)$ be a partially ordered set with a subset $A \subseteq V$ and an element $p \in V$. We define the following four concepts:

(1) p is a largest element (maximum) of A if $p \in A$ and each element of A is smaller than or equal to p.

(2) p is a smallest element (minimum) of A if $p \in A$ and p is smaller than or equal to any other element of A.

(3) p is a maximal element of A if $p \in A$ and each element of A is not strictly larger than p.

(4) p is a minimal element of A if $p \in A$ and each element of A is not strictly smaller than p.

The expressions (1) and (2) correspond well to our intuitive concept of 'largest' and 'smallest' element. The sentence of these properties are denoted in a more compact form in the following table.

#	Property	Definition
(1)	p is a largest element of A	$p \in A \wedge \forall a \in A, a \leq p$
(2)	p is a smallest element of A	$p \in A \wedge \forall a \in A, p \leq a$
(3)	p is a maximal element of A	$p \in A \wedge \forall a \in A, (p \leq a \rightarrow p = a)$
(4)	p is a minimal element of A	$p \in A \wedge \forall a \in A, (a \leq p \rightarrow a = p)$

The properties (3) and (4) may be a surprise in comparisons in verbal form.

Theorem 3.4. Given is a partially ordered set V with ordering $\leq$, and a subset A. The following holds: A cannot have two different largest elements and A cannot have two different smallest elements.

Proof. The arguments are the same for both statements; we, therefore, only prove the first statement. We formulate the goal as follows: if p and q are both largest elements of A, then $p = q$.

First, assume that p and q are both largest elements of A. The fact that p is a largest element, means that $p \in A$ and $\forall a \in A, a \leq p$. The fact that q is largest element, means that $q \in A$ and $\forall a \in A, a \leq q$. From $q \in A$ and $\forall a \in A, a \leq p$ it follows that $q \leq p$. From $p \in A$ and $\forall a \in A, a \leq q$ it follows that $p \leq q$. Now, from $p \leq q$ and $q \leq p$, we have $p = q$ (anti-symmetry). □

Theorem 3.5. Given is a partially ordered set V with ordering $\leq$, and a subset A. The following holds: a largest element of A is a maximal element of A, and a smallest element of A is a minimal element of A.

Proof. We only prove the first statement (the proof of the second statement is completely analogous). The goal is to show that: if p is the largest element of A, then p is a maximal element of A.

Assume that $p \in V$ and that p is a largest element of A. We will show that p is a maximal element of A. First, $p \in A$, since p is the largest element of A. Now, let $a \in A$ arbitrary and assume that $p \leq a$. Since p is the largest element of A, we have $a \leq p$. From $p \leq a$ and anti-symmetry of $\leq$, we conclude that $a = p$, as required for a maximal element. □

In case of a total ordering, the reverse implications also hold.

3.5 Exercises

1. In the zoo, the animals are fed according to a fixed schedule:

   (a) The giraffes are fed before the zebras, but after the apes.

   (b) The bears are fed immediately after the apes.

   (c) The lions are fed after the zebras.

   Can you recover the full schedule? Do you need additional 'common-sense' premises?

2. Let P be the set of all integers > 0; in the notation of sets: $P := \mathbb{N} \setminus \{0\}$. The relation IsDivisorOf (with symbol '$|$') in P is described as follows:

   $$x \mid y :\leftrightarrow \exists z \in P, x \cdot z = y.$$

(a) Show that '|' is a partial order on P.

(b) Show that the ordering is not total.

(c) Let $a = 12$ and $b = 30$. Show that the set

$$\{x : x \mid a \wedge x \mid b\}$$

of all common divisors x of a, b has a largest element in this ordering. Does this result also hold for arbitrary $a, b \in P$?

(d) Let $a = 12$ and $b = 30$ as in part (c). Show that the set

$$\{x : a \mid x \wedge b \mid x\}$$

of all common multiples x of a, b has a smallest element in this ordering. Does this result also hold for arbitrary $a, b \in P$?

3. Given is a set V with a total ordering $\leq$. Furthermore, it is given that $p \in V$ is a maximal element.

(a) Argue that p is a largest element of V.

(b) Give (for example through a Hasse diagram) an example of a partial (not total) ordering and an element that is maximal, but not the largest.

4. Let Alphabet be the set of alphabetic symbols 'a' ...'z' with the usual alphabetic ordering.

(a) Give (in words or with formulas) a description of the lexicographic ordering for the set Alphabet^4 of all four-letter words.

(b) Answer (a) for the Cartesian ordering for Alphabet^4.

(c) In this part Alphabet^4 has the lexicographic ordering. Let V be the set of all four-letter words with both at the second and third position one of the letters 'b' or 'c'. Give a largest and a smallest element of V or explain why such an element does not exist.

(d) In this part Alphabet^4 has the Cartesian ordering. Let W be the set of all four-letter words with at the second position a 'b' or at the third position a 'c' (or both). Give a largest and a smallest element of W or explain why such an element does not exist.

5. Take for V the set $\mathbb{Z} \times \mathbb{Z}$ with the Cartesian ordering (using the natural ordering on $\mathbb{Z}$), and for A the subset

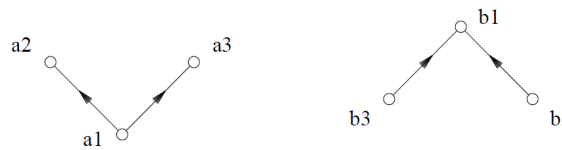
$$\{< x, y > : 1 \leq x, 1 \leq y, x + y \leq 5\}.$$

(a) Give a full enumeration of all elements of A and draw the Hasse diagram.

(b) Does A have a largest element? If so, please give this element; if not, give all maximal elements of A.

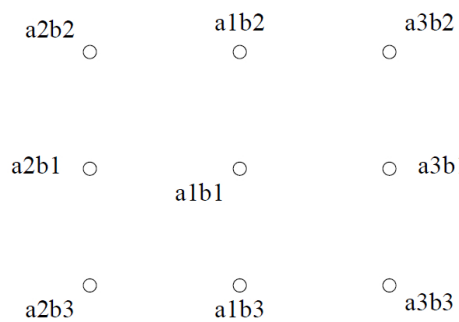
(c) Does A have a smallest element? If so, please give this element; if not, give all minimal elements of A.

6. Given are two sets $A := \{a_1, a_2, a_3\}$ and $B := \{b_1, b_2, b_3\}$, each with the partial order '$\leq$' described by the following Hasse diagrams.



In this exercise we study the Cartesian ordering of $A \times B$ (parts (a) and (b)) and the lexicographic ordering of $A \times B$ (part (c)).

(a) Develop a Hasse diagram for $A \times B$ with the Cartesian ordering. In the included figure all points are position such that the end result will be transparent.



(b) Give all maximal and minimal elements of $A \times B$ in the Cartesian ordering.

(c) The lexicographic ordering of $A \times B$ is not total. Give an example of two non-comparable elements in $A \times B$.

3.6 Background

The notion of a partially ordered set in its current form was formulated by Felix Hausdorff (1868–1942) in 1914. The three axioms (reflexivity, antisymmetry, and transitivity) were used earlier by Gottfried Wilhelm Leibniz (1646–1716) around 1690 and total orderings were used before by Georg Cantor in 1895. Hasse diagrams are named after the German number theoretician Helmut Hasse (1898–1979). For partial orderings that are not too large, this is a convenient representation.

Partial orderings are used in the study of correctness of computer programs (see [5]). But they also appear in problems such as topological sorting and job scheduling. For this last example, one needs a set of jobs that need to be processed to complete a project. The jobs can be ordered strict partially: $\text{job}_1 < \text{job}_2$ means that job_1 must be processed before job_2 can be started. In topological sorting, one needs to extend a partial order to a total ordering. For example, in job scheduling one has a partial order of jobs, that one tries to extend to a total ordering (the sequence of processing, see [4]).

Finally, boolean algebras can be seen as partially ordered sets with a largest element (usually named 1) and a smallest element (usually named 0). Additionally, there are two special properties. Among the common smaller elements of two given elements a, b, there is a largest (usually denoted by $a \wedge b$), and among the common larger elements of two given elements a, b, there is a smallest (usually denoted by $a \vee b$). Furthermore, for each element a there is an element a' for which $a \wedge a' = 0$ and $a \vee a' = 1$. This algebra operates in a similar way as the set-algebra and proposition-algebra. It is a generalization of both algebras.

Chapter 4

Relations: Equivalence

Goals of this chapter:

- Recognize an equivalence relation through its properties.
- Reproduce some relevant examples of equivalence relations (logic equivalence, congruence modulo m.
- Understand the connection between equivalence relations and a partition in equivalence classes.
- Recognize and design a complete system of representatives of an equivalence relation.

4.1 Refresher: binary relations

Basic concepts
In the previous chapter, we have again studied (binary) relations in a set V, i.e., relations of type $V \times V$. For binary relations, we use the infix notation $x\,R\,y$ to express that $< x, y > \in R$ ("x is in relation R with y").

Remember the definition of the following four properties that a relation R in a set V can have

Property	Formal definition
Reflexivity	$\forall x \in V, x\,R\,x$
Symmetry	$\forall x, y \in V, (x\,R\,y \rightarrow y\,R\,x)$
Anti-symmetry	$\forall x, y \in V, (x\,R\,y \wedge y\,R\,x \rightarrow x = y)$
Transitivity	$\forall x, y, z \in V, (x\,R\,y \wedge y\,R\,z \rightarrow x\,R\,z)$

In Chapter 3 we studied relations that satisfied a combination of three properties:

reflexivity + anti-symmetry + transitivity.

We called such a relation a partial order. In this chapter, we study a different combination of three properties:

reflexivity + symmetry + transitivity.

We change from anti-symmetry to symmetry. As will follow from the contents of this chapter, this change brings forth completely different behaviour of relations.

4.2 Equivalence relations

A relation R in a set V is called an equivalence relation when R satisfies the following three properties:

reflexivity + symmetry + transitivity.

The symbol '$\equiv$' is often used to denote an equivalence relation. Compare this with the use of '$\leq$' for orderings.

As we will show in this and next chapter, equivalence relations come from expressions of certain similarity of elements. Please pay attention to this phenomenon in the examples that we will discuss.

Example: Let Date_{10} be the set of all pairs $< x, y >$ of which x is an element of the set

DayNumbers := {01, 02, ..., 31},

and of which y is an element of the set

MonthNumbers := {01, 02, ..., 12},

such that the pair $< x, y >$ is a real date that occurs in 2010. As illustration: $< 31, 03 > \in \text{Date}_{10}$, but $< 31, 04 > \notin \text{Date}_{10}$. A date such as $< 31, 01 >$ is usually represented as 31-01. The relation R_{10} now exists of all pairs of dates that (in 2010) fall on the same day. For example, $< 31, 01 > R_{10} < 07, 11 >$, because both dates fall on a Sunday in 2010. One can easily check that the

defined relation R_{10} is an equivalence relation in the set Date_{10}. As a remark, note that this relation slightly depends on the chosen year, 2010. Can you describe the difference for this relation in the year 2008?

Let V be the Cartesian product $V_1 \times V_1$. For two arbitrary pairs $< x_1, x_2 >$ and $< y_1, y_2 >$ we describe the following relation:

$< x_1, x_2 > R < y_1, y_2 > :\leftrightarrow x_1 = y_1$.

Then R is an equivalence relation in V. In words: two pairs are equivalent if they have the same first coordinate. One can easily check that the described relation is an equivalence relation in $V_1 \times V_2$. An alternative equivalence relation is defined when we change the requirement to having the same second coordinate.

4.3 Two constructions of equivalence relations

First construction: logic equivalence of propositions
Logic equivalence in proposition logic is an important subject. Remember: two formulas (logic forms) F_1 and F_2 are called logically equivalent, symbolically: $F_1 \equiv F_2$, if they have the same truth tables. The use of the term 'equivalence' poses the question if equivalence relations have something to do with logic equivalence. This is indeed the case.

Examples:

(i) $\neg(p \vee q) \equiv \neg p \wedge \neg q$ (DeMorgan's Law)

(ii) $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$ (Distributive Law)

(iii) $(p \wedge \neg p) \vee q \equiv q$

In general, the formulas F_1 and F_2 contain several proposition symbols p_1, p_2, etc. Every row in the truth table of a formula has a certain value of truth (0 or 1) depending on the proposition, and give a truth value of the total formula. If $F_1 \equiv F_2$, then the outcomes on each row are the same. The computation of the truth value can be different; it is even possible that the formulas do not have exactly the same proposition symbols! One can see that in the example (iii) above.

p	q	$(p \wedge \neg p)$	$\vee q$	q
0	0	0	0	0
0	1	0	1	1
1	0	0	0	0
1	1	0	1	1

The column of q is (superfluously) repeated as outcome of the second formula. The outcomes in this column are $0-1-0-1$, of which the first two belong to the value 0 of p, and the next two are a repetition of the previous outcomes for the value 1 of p: this value does not matter since p does not appear in the formula q. The final values of both formulas $(p \wedge \neg p) \vee q$ and q (denoted in bold) are the same for every line.

Back to the general case. We list the relevant properties of logic equivalence:

Reflexivity: A formula has the same truth table as itself.

Symmetry: If F_1 has the same truth table as F_2, then clearly, F_2 has the same truth table as F_1.

Transitivity: First collect all proposition symbols that appear in F_1, F_2, and F_3. Next, create a joint truth table for the three formulas. If F_1 has the same truth value as F_2 on each line, and if F_2 has the same truth value as F_3 on each line, then F_1 has the same truth value as F_3 on each line.

Logic equivalence also appears in sentences (predicate logic). One can easily check that also here an equivalence relation appears.

Second construction: congruence modulo an integer
The second construction pertains arithmetic modulo an integer. Such systems of arithmetic appear surprisingly often in daily life. In all of the following examples, a special number m, the modulus, is present.

- Read a clock: the hours on a clock run constantly in a cycle of 12 or 24. If we have a 12-hour clock, then we do not say that 'it is 15 'o clock', but we say 'it is 3 'o clock'. The number 3 is the reduction of 15 modulo 12 ($m = 12$). If we have a 24-hour clock, then we do not say 'it is 29 'o clock', but we say 'it is 5 'o clock'. The number 5 is the reduction of 29 modulo 24 ($m = 24$).

- Arithmetic with minutes and seconds: minutes and seconds constantly run in cycles of 60. We do not say that "this movie lasts 2h77min', but 'this movie lasts 3h17min'. The number 17 is the reduction of 77 modulo 60 ($m = 60$). Something similar can be said of seconds.

- Arithmetic with weekdays: weekdays constantly run in a cycle of 7. If it is Tuesday today, than 8 days later it is Wednesday, this is one day further in the cycle ($m = 7$).

- Shifts in the alphabet: simple cryptographic systems render a text unreadable by shifting the letter in the alphabet by a fixed number (the so-called Caesar code). The letters at the end that tend to be discarded, are put at the beginning of the alphabet. This corresponds to arithmetic modulo $m = 26$.

  More precisely, we represent 0 for 'a', 1 for 'b', 2 for 'c', etc. until 25 for 'z'. Assume that we want to shift three letters. The letter 'y' is represented by the number 24. If the shift is applied, then we get 27. This contains a full cycle of 26 plus 1. This is the code for the letter 'b'.

- Number representation in a computer: ($m = 2^{32}$). Most modern computers work with units of memory of 32 bits (or 4 bytes). One can quickly see that every number from 0 to $2^{32} - 1 = 4,294,967,295$ can be represented. In order to add to number, the binary representation with bitsum and carry are used:

27	→	00000000000000000000000000011011
55	→	00000000000000000000000000110111
+		+
82	→	00000000000000000000000001010010

1 294 967 296	→	01001101001011111010001000000000
3 000 000 001	→	10110010110100000101111000000001
+		+
4 294 967 297	→	00000000000000000000000000000001

  The binary scheme is an obvious adjustment to the decimal method. The number 2 plays the role of a ten-fold. When the outcome of an addition is greater than $m - 1 = 4294967295$, then the first bits appear

to be discarded. In more imaginary terms: the bit-reservoir has an overflow on the left side. Instead of a really large number, we only end up with the overflow, the part that was greater than m.

Arithmetic modulo m

What does one exactly mean when one does arithmetic modulo a number m. For a given integer $m \geq 2$, we focus on the relation R with description

$x \, R \, y \quad :\leftrightarrow \quad y - x$ is divisible by m.

We shall verify that this relation in an equivalence relation in $\mathbb{Z}$. It is also called the congruence modulo m and the number m is called to modulus (moduli in plural). Instead of $x \, R \, y$ one writes

$x \equiv y \pmod{m}$.

The addition of "mod m" to the equivalence symbol '$\equiv$' reminds us to the used modulus m.

Examples:

$17 \equiv 5 \pmod{12}$, because $5 - 17 = -12$, which is divisible by 12.
$11 \equiv 25 \pmod{7}$, because $25 - 11 = 14$, which is divisible by 7.
$-3 \equiv 181 \pmod{2}$, because $181 - (-3) = 184$), which is divisible by 2.

For each integer x, one can find a remainder r after integer division of x by m. If q is the quotient, then $x = mq + r$ and $0 \leq r < m$. Since, $x - r = mq$ if follows that r is congruence modulo m with x. One also called r the reduction of x modulo m. A number is, thus, always congruent modulo m with his reduction modulo m.

Example: the reduction modulo 24 of the number 51 is 3. The reduction modulo 10 of a natural number is the last digit of that number.

We list the relevant properties of 'congruence modulo m' for a given modulus $m \geq 2$:

Reflexivity: $x \equiv x \pmod{m}$ because $x - x = 0$ is divisble by m.

Symmetry: Assume that $x \equiv y \pmod{m}$. Then, $x - y$ is divisible by m. But then also reversely, $y - x$ is divisible by m. Hence, $y \equiv x \pmod{m}$.

Transivity: Assume $x \equiv y \pmod{m}$ and $y \equiv z \pmod{m}$. Then $x - y$ is divisible by m, and $y - z$ is divisible by m. But then the sum of these numbers is divisible by m. This sum is:

$(x - y) + (y - z) = x - z$.

Consequently, $x \equiv z \pmod{m}$. We conclude that the relation 'congruence modulo m' is an equivalence relation in $\mathbb{Z}$.

As a side remark: why do we restrict ourselves to moduli m with $m \geq 2$? It is clear that $m = 0$ is not valid, due to division by zero. If one tries $m = 1$, then it is not a useful relation: the trivial relation $\mathbb{Z} \times \mathbb{Z}$. For positive m one automatically gets $m \geq 2$. And how about a negative value for m? Simply put: divisibility by m is the same as divisibility by $-m$. Hence, we can use the positive modulus in future.

4.4 Equivalence classes and partitions

Equivalence classes

Let R be an equivalence relation in a set V and let $p \in V$. We define the equivalence class of p with respect to R (or shorter: the R-equivalence class of p, or even shorter, the class of p) as the set

$[p] := [p]_R := \{x : x \in V, p\,R\,x\}$.

Hence, the equivalence class of p exists exactly of the elements of V which p is related to in R. The reference to the equivalence relation R is usually omitted.

Example: Consider the equivalence relation "congruence modulo 2" in $\mathbb{Z}$. The class of $0 \in \mathbb{Z}$ is according to the definition

$[0] = \{x : 0 \equiv x \pmod{2}\} = \{x : 2|0 - x\}$,

en thus $[0]$ is the set of all even numbers. Similarly,

$[1] = \{x : 1 \equiv x \pmod{2}\} = \{x : 2|1 - x\}$,

and thus $[1]$ is the set of all odd numbers.

Example: The cube $\{0,1\}^3$ exists of all triples of bits (0/1). An equivalence relation R in the cube can be defined by the following description:

$w_1\,R\,w_2$ exactly then if w_1 and w_2 have an equal number of zeros (and thus an equal number of ones as well).

The equivalence classes of R are:

$[000] = \{000\}$	$[111] = \{111\}$
$[001] = \{001, 010, 100\}$	$[110] = \{110, 101, 011\}$
$[010]$ = the same as $[001]$	$[101]$ = the same as $[110]$
$[100]$ = the same as $[001]$	$[011]$ = the same as $[110]$.

Note that there are exactly four different classes.

Theorem 4.1. Let R be an equivalence relation in a set V, and let p and q be two elements of V. The following statements are true:

(i) $p \in [p]$.

(ii) $[p] \cap [q] \neq \emptyset \rightarrow [p] = [q]$.

Proof. The statement that $p \in [p]$ follows directly from reflexivity of R: $p\,R\,p$. We therefore focus on the second part.

Assume that $[p] \cap [q] \neq \emptyset$. Then, there is an element $v \in [p] \cap [q]$. From $v \in [p]$ it follows that $p\,R\,v$. From $v \in [q]$ it follows that $q\,R\,v$, and thus $v\,R\,q$ (by symmetry of R). From $p\,R\,v$ and $v\,R\,q$ it follows that $p\,R\,q$ (by transitivity of R). We can show that $[q] \subseteq [p]$. Take $x \in [q]$ arbitrary. Then (by definition) $q\,R\,x$, while we know that $p\,R\,q$. Thus (by transitivity) $p\,R\,x$, i.e., $x \in [p]$. In a completely analogous manner, one can show that $[p] \subseteq [q]$. With both inclusions, we have that $[p] = [q]$. □

Remember that a partition $\mathcal{P}$ of a set V is a collection of non-empty subsets of V, such that each element of V is member of exactly of subset. We can express this in the language of predicate logic as follows:

(i) $\forall P \in \mathcal{P}, (P \subseteq V \wedge P \neq \emptyset)$.
(ii) $\forall x \in V, \exists! P \in \mathcal{P}, x \in P$.

An alternative formulation is the following: a partition of a set V is a collection of non-empty, pairwise disjoint subsets of V, of which the union is equal to V. We can express this in the language of predicate logic as follows:

(i) $\forall P \in \mathcal{P}, (P \subseteq V \wedge P \neq \emptyset)$.
(ii) $\forall P_1, P_2 \in \mathcal{P}, (P_1 \neq P_2 \rightarrow P_1 \cap P_2 = \emptyset)$.
(iii) $\forall x \in V, \exists P \in \mathcal{P}, x \in P$.

Both definitions say the same thing (this is left as an exercise to the reader to check this).

Theorem 4.2. Let R be an equivalence relation in a set V. Then, the equivalence classes of R form a partition of V.

Proof. The equivalence classes of R are non-empty subsets (the class of an element p contains at least p). Let x be an arbitrary element of V. There is at least one equivalence class that contains x, namely $[x]$. There is also at most one equivalence class that contains x, because if $[p]$ and $[q]$ both contain the element x, then $[p] = [q]$ according to Theorem 4.1. □

Example: The different equivalence classes of the equivalence relation "congruence modulo 2" are: $[0]$ and $[1]$. These sets form the partition "even versus odd" numbers in $\mathbb{Z}$. More generally, if $m \geq 2$, then $[0], [1], \ldots, [m-1]$ are the different equivalence classes for congruence modulo m. The fact that classes $[p]$ and $[q]$ $(0 \leq p \neq q \leq m-1)$ are different, follows simply from the fact that $p - q$ cannot be divisible by m, because $-(m-1) \leq p - q \leq m-1$ and $p - q \neq 0$. Also, this set represents all classes and this follows from the possibility of reduction modulo m: for given $a \in \mathbb{Z}$, the remainder of a divided by m is a number r congruent to a modulo m, where $0 \leq r \leq m-1$. As a consequence, $[a] = [r]$ is one of the mentioned classes. The resulting partition $\mathcal{P}$ consists of the set of multiples of m, the set of multiples of m plus one, etc., and the set of all multiples of m plus $m-1$.:

$\mathcal{P} := \{[0], [1], \ldots, [m-1]\}$.

Example: Remember the equivalence relation R in the cube with the description

$w_1 \, R \, w_2$ exactly then if w_1 and w_2 have an equal number of zeros (and thus an equal number of ones as well).

There are exactly four different equivalence classes in R:

$\{000\}, \{111\}, \{001, 010, 100\}, \{110, 101, 011\}$.

These sets clearly form a partition of the elements of the cube.

Equivalence relations from partitions

In Theorem 4.2, it was shown that an equivalence relation gives rise to a partition of equivalence classes. We now show that the reverse also holds.

Theorem 4.3. Let $\mathcal{P}$ be a partition of the set V. The description

$x \, R \, y :\leftrightarrow \exists P \in \mathcal{P}, (x \in P \wedge y \in P)$,

is an equivalence relation in V. The equivalence classes of this relation are exactly the members of the partition.

Proof. We first check that the relation is an equivalence relation.

- The relation is reflexive: Let $x \in V$ be arbitrary. Then there exists a set $P \in \mathcal{P}$ with $x \in P$. We find $x \, R \, x$ according to the description.

- The relation is symmetric: this is a direct consequence of commutativity of '$\wedge$'.

- The relation is transitive: Let $x, y, z \in V$ arbitrary and assume that $x \, R \, y$ and $y \, R \, z$. Because of $x \, R \, y$, there exists a set $P_1 \in \mathcal{P}$ with $x, y \in P_1$. Because of $y \, R \, z$, there exists a set $P_2 \in \mathcal{P}$ with $y, z \in P_2$. Apparently, y belongs to two sets P_1 and P_2 of a partition. According to the definition these sets are the same: $P_1 = P_2$. Hence, there is a set that simultaneously contains x and z. Therefore, $x \, R \, z$.

We finally show that the equivalence classes of R are exactly the partitions of $\mathcal{P}$.

For a given equivalence class $[v]$ with $v \in V$, there exists a $P \in \mathcal{P}$ with $v \in P$. We check that $P = [v]$. Every element of P satisfies per definition of R the condition $v \, R \, x$. This shows that $P \subseteq [v]$. Reversily, every $x \in [v]$ satisfies per definition of an equivalence class the condition $v \, R \, x$. According to the definition of R, there is a $P' \in \mathcal{P}$ with $x \in P'$ and $v \in P'$. Since P is unique with $v \in P$, we have $P' = P$ and consequently $x \in P$. Hence, we have shown that $[v] \subseteq P$.

For a given partition $P \in \mathcal{P}$, there exists an element $v \in P$; we need to check that $[v] = P$. This can be done similar to the proof above. □

Example: The set $\{0, 1\}^3$ of all lists of bit of length three can be split into four sets: the list without zeros, the list with one zero, the list with two zeros, and the list with three zeros. This leads to the equivalence relation "have an equal number of zeros" in the set of lists of bits of length three.

Example: Consider the set Date_{10} of all real dates in 2010. We can divide this set of 365 elements into seven parts, of which each part consists of all dates that fall on a particular day (Sunday, Monday, etc.). This partition leads to the equivalence relation "falls on the same day".

Complete system of representatives
Let V be a set and R an equivalence relation in V. A set $S \subseteq V$ is called a complete system of representatives of R if S contains on element from each R-equivalence class. Note that a complete system of representatives contains as many elements as there are equivalence classes.

Example: Consider the equivalence relation in Date_{10}, in which two dates are in relation when the fall on the same day (Sunday, Monday, …) in 2010. We conclude that 1 January 2010 is a Friday, 2 January 2010 a Saturday, 3 January a Sunday, 4 January a Monday, 5 January a Tuesday, 6 January a

Wednesday, and 7 January a Thursday. A complete system of representatives is there, e.g.,

$\{< 01,01 >,< 02,01 >,< 03,01 >,< 04,01 >,< 05,01 >,< 06,01 >,< 07,01 >\}$.

However, one can take any other seven consecutive dates in 2010: these will also form a complete system of representatives.

Example: If $m \geq 2$, then the complete system of representatives of congruence modulo m is given by $0, 1, \ldots, m-1$. This is a consequence of the theorem that $[0], [1], \ldots, [m-1]$ is the collection of all equivalence classes. The set $\{0, 1, \ldots, m-1\}$ has one representative out of these equivalence classes.

Theorem 4.4. Let V be a set and R an equivalence relation in V. A set $A \subseteq V$ is a complete system of representatives of R if and only if the following two conditions hold.

(i) Each element of V is equivalent to an element of A, in logical form $\forall v \in V, \exists a \in A, v\, R\, a$.

(ii) Two different elements of A are not equivalent with respect to R, in logical form: $\forall a_1, a_2 \in A, (a_1 \neq a_2 \rightarrow \neg a_1\, R\, a_2)$.

Proof. First assume that A is a complete system of representatives. of R, and take an arbitrary element of $v \in V$. Then, $[v]$ is an equivalence class, and by assumption, it contains a element $a \in A$. For this a we find $v\, R\, a$ which shows (i). Now assume that $a_1 \neq a_2$ in A and assume that $a_1\, R\, a_2$. Then, a_1 and a_2 are two different representatives of the same equivalence class; this is in contradiction with the assumption on A. Hence, this proves (ii).

Reversely, let A satisfy the conditions (i) and (ii). We show that A is a complete system of representatives of R as follows. Take an arbitrary equivalence class of R. This is of the form $[v]$ for certain $v \in V$. This class has a representative in A because of (i), and it cannot have two representatives in A because of (ii). □

4.5 Exercises

1. Let E be the set of all non-empty (letter)words of length smaller than or equal to 6. Two words w_1 and w_2 in E are in relation R if they have the same length and if they both begin with a vowel or they both begin with a consonant.

(a) Show that R is an equivalence relation in E.

(b) How many different equivalent classes are represented by the following words:

"yes", "no", "indeed", "not", "some", "none", "chem", "ai", "inf", "imm".

(c) Give a complete system of representatives for the relation R in E.

2. Given is a set Throws := Number × Number, in which Number := {1, 2, ..., 6}, together with a collection of 11 subsets $P_2, \ldots, P_{12}$. Set number k (with $2 \leq k \leq 12$) is defined as:

$$P_k := \{< o_1, o_2 > : o_1, o_2 \in \text{Number}, o_1 + o_2 = k\}.$$

Note that this collection of 11 subsets is a partition of the set Throws.

(a) Give a convenient description of the corresponding equivalence relation in Throws.

(b) Design a complete system of representatives for the equivalence relation of part (a).

3. Check for each of the following relation if they are an equivalence relation in the specificied set. Please give a complete and convenient system of representatives.

(a) The relation $x = y$ in the set $\mathbb{N}$.

(b) $< a, b > \equiv < c, d > :\leftrightarrow a + d = b + c$ in the set $\mathbb{N} \times \mathbb{N}$.

(c) $< a, b > \equiv < c, d > :\leftrightarrow a \cdot d = b \cdot c$ in the set $\mathbb{Z} \times (\mathbb{Z} \setminus \{0\})$ (pairs of integers with the second coordinate different from zero).

(d) Let P be the set of all functioning programmes that each accept as input a word of 32 bits and return a word of 32 bits as output. Define the relation R in P with the following description:

$x\,R\,y$ means that for every acceptable input both programmes x and y deliver the same output.

4. In $\mathbb{R}^2$ we have a partition by sets of the form

$$C_r := \{< x, y > : x^2 + y^2 = r^2\},$$

with $r \geq 0$ is a real number. Note that C_r can be represented by a circle around the origin with radius r. For $r = 0$ the circle is "degenerate" and yields a single point. Provide a convenient description for the corresponding equivalence relation in formulas. Is there a complete and convenient system of representatives?

5. Given is a set V and a partition $\mathcal{P}$ of V. Provide a convenient description for the equivalence relation in V that comes forth from $\mathcal{P}$ in the following cases.

   (a) V is the set of (alphabetic) words and $\mathcal{P}$ consists of sets $P_n := \{x : x \text{ has length } n\}$ for $n = 0, 1, \ldots$.

   (b) $V := \mathbb{R}$ and $\mathcal{P}$ consists of sets $P_z := \{x : \lfloor x \rfloor = z\}$ for $z \in \mathbb{Z}$. Note that $\lfloor x \rfloor$ is the entier of $x \in \mathbb{R}$, i.e., it is a largest integer z for which $z \leq x$ ("rounding down").

4.6 Background

Equivalance relations are, next to order relations, the most frequently used class of binary relations. The view on logic equivalence via truth tables with the same values has a parallel in the important concept of equivalence in automata. These abstract machines can indeed be seen as a "behaviour table" with an output for a given input and a state.

The relation "congruence modulo m" was already studied by one of the most prominent mathematicians of all times: Carl Friedrich Gauss (1777-1855). Modulo arithmetic is the basis of modern number theory and her many applications in (among others) cryptography and coding. See references [1] and [12] for an elementary introduction to this subject.

Equivalence classes are used as a tool for certain techniques to test software. The underlying concept is to classify potential input data into specific data classes, in which elements of the same class are treated in similar way by the program [8].

Chapter 5

Functions

Goals of this chapter:

- Recognize functions in binary relations.
- Has knowledge of special properties of functions: partial versus total, surjective, injective (one-one), and bijective.
- Can determine the domain and the codomain of a function.
- Can connect the concepts between partitions in equivalence classes in equivalence relations (quotient set) and similarity in elements under a surjective function.
- Determine the composition of two functions, and check if the inverse function is a function as well.
- See the relation between counting processes and bijections of finite sets, and the extension to infinite sets.
- Knows the cardinality of infinite sets and the gradations of infinity, among which the concept of countability.

Remark on terminology. What we call total functions in these notes, are simply called functions in many other texts; consequently, what we call functions in these notes, are then called partial functions in those texts.

5.1 Processes and functions

Computer produce output for a given input. Between the input and the output, there is a process that translates the first into the latter. This process has some special properties: it is steered by precise instructions and rules (the process is algorithmic and it produces unambiguous results (the process is deterministic. These two properties usually go hand in hand.

Example: Consider the process in which a text in Dutch is check for spelling errors. The input is: a Dutch text. The output is: a list of incorrectly spelled words in the text. The process description: A list is created of all consecutive words in the text. Each word that is read and that is already in the list is ignored. The resulting list is compared word for word with a standard list (a dictionary) taking into account different forms of the words (singular or plural forms, etc). Remove the word from the list if it is recognized in the list. The list of words that remain is the output.

We have left out most technical details of this usual procedure. The mentioned characterics (precision, unambiguity) are visible even without the details.

Example: Instructions such as "divide the number by two or multiply the number with three", or "double the number several times" are not precisely formulated (for a given input) and therefore the output is undetermined.

Example: A cooking book typically contains examples of processes that are not (or partly) algorithmic / deterministic in nature. Think of instructions as

"Leave the dish cooking for several minutes on the stove",
"Add some spices according to your taste".

Give the input (the ingredients of the recipe), the output (the final dish) can strongly vary based on the person that has prepared it.

Functions: informal
Some buttons of your calculator (such that the button for the square root, sine, cosine, etc.) are also called function buttons. After input of a specific number and pressing of the button, an unambiguous answer appears on screen after running a process, of which one does not need to know the details (and usually does not know it either).

The central concept in this chapter is based on this phenomenon. A

function is the relation between input and output of a deterministic process. The process can be seen as a black box. With the concept of a function, we do not focus on the details of the implementation or formula, but we focus on the outcome of it.

Example: square root
Input: a positive real number x.
Output: $\sqrt{x}$.
Process: a technical procedure to calculate the square root.
The corresponding function ties $\sqrt{x}$ to x.

Example: reverse a sequence
Input: a word (list of letters) w.
Output: the reverse of the word w.
Process: a technical procedure to reverse the letters in a word.
The corresponding function ties the reverse of a word to itself.

Example: read a table
The process behind a function usually consists of reading a table with two rows. The first row consists of all acceptable input values, the second row is the corresponding output value for the input value. Here is an example. In the jargon of mountain climbers, one speaks of a 'summit' when someone reaches the top. The following table lists a few summits for the Mount Everest from 1985 to 1997. The input is the year, and the output is the number below in the table. For example: in 1986 there were 4 people that reached the top of the Everest.

Year:	85	86	87	88	89	90	91	92	93	94	95	96	97
Summits:	30	4	2	50	24	72	38	90	129	51	83	98	85

5.2 Mathematical description

Functions: formal
In order to describe functions informally, we did not bother with the details of the process that leads from input to output. What remains is: a binary relation between possible input and output, where an unambiguous output value is coupled to an acceptable input. Now that we have all key ingredients for a function, we can give a formal mathematical definition.

A function is a binary relation of which each element is in relation with at most one element. Formally expressed: a function is a binary relation f

such that

$$\forall x \forall y_1 \forall y_2, (x\, f\, y_1 \wedge x\, f\, y_2 \rightarrow y_1 = y_2).$$

This expresses quite nicely the concept of determinism.

Such as with sets and relations, a function can be given a name. The name can be chosen suggestively (e.g., square-root, summits). In a general context, we usually use the neutral names such as f, g, ….

Typing

For a binary relation R one usually gives the type $A \times B$. We say that R is of type $A \times B$. This expresses that $R \subseteq A \times B$. Explicitly: if $x\, R\, y$, then $x \in A$ and $y \in B$. Since functions are binary relations, we can also provide a type for functions. We can then speak of typed functions. A function of type $A \times B$ is also described as a function from A to B. We denote this in the following way:

$$f : A \rightarrow B, \quad A \xrightarrow{f} B.$$

The set A is called the domain and B the codomain of the function f.

Domain of definition and image

Every function has a domain of definition: this is the set of all acceptable inputs of the function. The domain of definition of a function f is denoted by D_f. Note the difference: 0 is a possible input of the real-valued function $1/x$ of type $\mathbb{R} \rightarrow \mathbb{R}$, but it is not an acceptable input and does not belong to the domain of definition of this function.

As a side remark: It may be strange on first sight that next to the domain of definition D_f we also have the domain A. The distinction is convenient, for example, with programming. When defining a procedure, one can use A for the type of the input variable (assume, e.g., integer) and D_f can be the set of integers that are really accepted by the procedure. In case of programming the square root procedure, the set A consists of integers, and D_f consists of the set of non-negative integers. When choosing a data-type for A, one chooses for D_f the set of "real" input. The benefit is that A can in principle be checked at compile time, and D_f can sometimes only be checked at run time.

A function also has an image: this is the set of all producible outputs. In general, we rather speak of values than the output of a function. The image of f is denoted by W_f. Note the difference: -1 is element of the codomain of the real-valued function x^2 of type $\mathbb{R} \rightarrow \mathbb{R}$, but is not effectively assumed as value.

Examples: Remember the functions square root and summits.
$D_{\text{square root}} = \{x : x \in \mathbb{R}, x \geq 0\}$,
$W_{\text{square root}} = \{x : x \in \mathbb{R}, x \geq 0\}$,
$D_{\text{summits}} = \{85, 86, \ldots, 97\}$,
$W_{\text{summits}} = \{2, 4, 24, 30, 38, 50, 51, 72, 83, 85, 90, 98, 129\}$.

Value and argument
The value of a function f for acceptable input x is usually denoted as $f(x)$ or $f\,x$. When the input x is transformed into a value y by a function f, we can denote this in the following ways.

$$f : x \mapsto y, \quad x \overset{f}{\mapsto} y, \quad f(x) = y.$$

In the expression $f(x)$ we call x the argument of the function f. We will preferably use the term argument instead of input in the sequel. Similarly, we will use value instead of output.

Examples:
$\text{square root}(100) = 10$; $\quad 100 \overset{\text{square root}}{\mapsto} 10$,
$\text{summits}(97) = 85$, $\quad 97 \overset{\text{summits}}{\mapsto} 85$.

Relations are often defined by a description. This can also be done for defining functions by so-called function descriptions.

The function add2 of type $\mathbb{N} \to \mathbb{N}$ has description $x \mapsto x + 2$. We can rewrite the descriptions also as $\text{add2}(x) := x + 2$. The function NextLetter of type Alphabet $\to$ Alphabet has as description: NextLetter(x) is the letter of the alphabet that immediately follows x.

Partial function versus total function; surjective function
For a typed function $f : A \to B$ each acceptable input comes from the domain A and each producible output comes from the codomain B. In set-theoretic language, this is represented as

$$D_f \subseteq A, \quad W_f \subseteq B.$$

We speak of a total function $f : A \to B$ when the domain and the domain of definition of f coincide: $D_f = A$. Explicitly: a (typed) function is total if every element of the domain is also effectively accepted as input of the function. To stress the contrast, we shall speak of a partial function in the general case. Compare the language with the partial orderings.

There is also a specific term for a typed function $f : A \to B$ for which

the codomain is equal to the image: $W_f = B$. Such functions are called surjective. In words: a (typed) function is surjective if every element of the codomain is effectively produced as a value of the function.

Example: Consider the function square root with domain of definition the real numbers $x \geq 0$. Indeed, only such numbers have a (real) root. The value $\sqrt{x}$ is a number ≥ 0. However, the result of the square root is also a number ≥ 0. Hence, the image is also equal to $\{x : x \in R, x \geq 0\}$.

The function square root can also be typed as $\mathbb{R} \to \mathbb{R}$. When typed in this way, the function is not total since only non-negative numbers are accepted as input, and not surjective since only non-negative numbers are the result of the square root.

Example: Take the function f that produces $1/x^2$ for every number x. A fitting type is $\mathbb{R} \to \mathbb{R}$. The domain of definition of f consists of all real numbers different from 0, thus f is not total. The image of f consists of all real numbers greater than 0. The function f is therefore not surjective either.

Let V_1 and V_2 be non-empty sets and let V be the Cartesian product $V := V_1 \times V_2$. We can now construct a function f that for a given pair from V returns the first coordinate: thus, the input $< v_1, v_2 >$ yields as output v_1 where $v_1 \in V_1$ and $v_2 \in V_2$. The domain of definition of f is the whole Cartesian product $V_1 \times V_2$. The image of f is the first factor V_1 of the product. This function f is also called the projection on the first coordinate. The type is $V_1 \times V_2 \to V_1$. Instead of the first coordinate, one can also take the second coordinate. In this way, one creates a projection on the second coordinate with type $V_1 \times V_2 \to V_2$.

The previous construction can be generalized to a Cartesian product of n non-empty sets ($n \geq 2$):

$V := V_1 \times V_2 \times \cdots \times V_n$.

One obtain for each $i = 1, 2, \ldots, n$ the i-th projection of the product V on the i-coordinate V_i. The i-th projection is of type $V \to V_i$. The domain is the Cartesian product V and the codomain is the i-th coordinate V_i. The i-th projection is a function with domain of definition V and image V_i. Hence, this function is both total and surjective. (Check yourself what goes wrong when one of the set $V_1, \ldots, V_n$ is empty).

Projections of a product set $V_1 \times V_2 \times \cdots \times V_n$ on the factors $V_1, V_2, \ldots, V_n$

are examples of what is called: a function with n arguments. In general, there are also function of type

$$V_1 \times V_2 \times \cdots \times V_n \to W.$$

The input is an n-pair $< x_1, x_2, \ldots, x_n >$ with $x_i \in V_i$ for $i \in \{1, 2, \ldots, n\}$. For such an input, x_i is called the i-th argument of the function. When f is the name of the function, one rewrites the correct $f(< x_1, \ldots, x_n >)$ as the short convenient $f(x_1, \ldots, x_n)$.

Remark: terms in the predicate logic. In predicate logic, one sometimes use terms such as "father of …" en "the double of …". In the formal language, these were represented in this style: $t(x)$. The suggestion that this is a function is correct. More precisely: the interpretation of a term (with one argument) in a universe U is a function in that universe. It is also a total function $U \to U$, because a term $t(x)$ should have a subject for each value of x. A term with n arguments corresponds to a total function $U^n \to U$.

5.3 Functions and equivalence relations

Refresher
An equivalence relation R in a set V is reflexive, symmetric, and transitive. For a given relation R in V, we can determine the equivalence class $[v]$ of an element $v \in V$ as the set

$$[v] := \{x : v\, R\, x\}.$$

These classes together form a partition $\{[v] : x \in V\}$ of V. This means that the classes are non-empty, pairwise disjoint, and together they contain all elements of V.

Quotient set and the quotient function
The partition that belongs to an equivalence relation R in a set V is sometimes denoted by V/R, read: V modulo R. Hence, the quotient set V/R is the set of all R-equivalence classes:

$$V/R := \{[x] : x \in V\}.$$

We now can introduce a function V to V/R by defining

$$q : V \to V/R, \quad q(x) := [x].$$

The process that belong to q is thus the construction of the equivalence class of x. This function is called to quotient function corresponding to the equivalent relation R.

Example: Take the equivalence relation "congruence modulo m". This relation is denoted symbolically (m). The corresponding partition is then $\mathbb{Z}/(m)$. One simply reads this as "$\mathbb{Z}$ modulo m". We have already shown that

$$\mathbb{Z}/(m) = \{[0], [1], \ldots, [m-1]\}.$$

Arithmetic modulo m actually takes place in this set, the number system modulo m. We also have a quotient function q of type $\mathbb{Z} \to \mathbb{Z}/(m)$ that converts integers into "numbers modulo m". For example:

$$q(0) = \{0, m, -m, 2m, -2m, \ldots\} =: m\mathbb{Z},$$
$$q(1) = \{1, m+1, -m+1, 2m+1, -2m+1, \ldots\} =: m\mathbb{Z}+1.$$

Theorem 5.1. Let R be an equivalence relation in V, and q the quotient function corresponding to R. Then, q is a total function:

$$\forall x \in V, \forall y \in V, (x\,R\,y \leftrightarrow q(x) = q(y)).$$

Proof. The fact that q is a total function is given by the fact that each element x of V has an R-equivalence class $[x]$. We concentrate our attention to the formula.

Take $x, y \in V$ arbitrary, and assume that $x\,R\,y$. We want to show that $q\,x = q\,y$. Because $x\,R\,y$, we have $y \in [x]$ and $x \in [y]$ (by definition of equivalence class). Hence, (consequence of Theorem 4.1 of Chapter 4) we have $[x] = [y]$. Thus, $q\,x = q\,y$.

Now assume that $q\,x = q\,y$. We want to show that $x\,R\,y$. From $q\,x = q\,y$ we have $[x] = [y]$. Since $y \in [y]$, it follows that $y \in [x]$ and thus (by definition of equivalence class) $x\,R\,y$. □

Theorem 5.2. Let $f : V \to W$ be a total function. The relation R with description

$$x\,R\,y :\leftrightarrow f(x) = f(y)$$

is an equivalence relation in V.

Proof. Checking reflexivity and symmetry is left as an exercise to the reader. We prove the transitivity of R.

Let x, y, z in V be arbitrary elements and assume that $x\,R\,y$ and $y\,R\,z$. According to the definition, we have $f\,x = f\,y$ and $f\,y = f\,z$ and therefore $f\,x = f\,z$. Again, by using the definition, we have $x\,R\,z$. □

Theorem 5.2 says that a description $f(x) = f(y)$ of similarity leads to an equivalence relation. Theorem 5.1 says the reverse: an equivalence relation leads to a description of similarity through a specific function. In both theorems, the function are total.

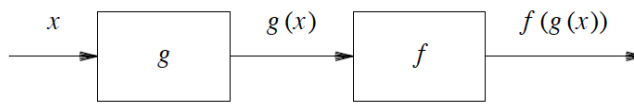
Example: We consider the set $\mathbb{Z}$ of integers and a number $m \in \mathbb{N}$ with $m \geq 2$. Then, for every $x, y \in \mathbb{Z}$ we have $x \equiv y \pmod{m}$ if and only if the remainder of x after division by m is the same as the remainder of y after division by m. In this example the description of similarity is given by a function f of type $\mathbb{Z}to\mathbb{N}$ with

$f(x) :=$ remainder of x after division by m.

5.4 Composition and inverse

Composition of functions

Functions are special relations and can thus be composed just like relations. The result is similar to the result of running two processes after each other. Each of the processes give an output for a given (acceptable) input. We called this 'determinism' before. It is not difficult to see that the composite process is also deterministic. The figure that belong to the composition $f \circ g$ (f after g) is as follows.



The description corresponding to the composition $f \circ g$ of two function f and g can also depicted by

$$x \overset{g}{\mapsto} y \overset{f}{\mapsto} z, \qquad x \overset{g}{\mapsto} g(x) \overset{f}{\mapsto} f(g(x)).$$

As a side note: the "$\circ$" symbol was originally developed to depicted compositions of functions. The notation was later generalized for compositions of two relations.

In the composition of $f \circ g$ is can happen that a certain output of g is not acceptable as input for f. In many textbooks two functions f and g can also be composed when the condition $W_g \subseteq D_f$ is met. In words: each output of g is an acceptable input of f. This relates to the usual condition that the domain and the domain of definition of a typed function should be the same. This restriction is not necessary, but is a tradition that is of no use in this course.

Example: Take two real-valued function f and g with description

$$x \overset{g}{\mapsto} 1 - x^2, \qquad x \overset{f}{\mapsto} \sqrt{x},$$

respectively. The image of g and the domain of definition of f are as follows:

$$W_g = \{x : x \leq 1\}, \qquad D_f := \{x : 0 \leq x\}.$$

Note that $W_g \not\subseteq D_f$. Furthermore, $f \circ g(x)$ is given by $\sqrt{1-x^2}$. The domain of definition of this composition is given by

$$D_{f \circ g} = \{x : 0 \leq 1 - x^2\} = \{x : -1 \leq x \leq 1\}.$$

One-to-one and many-to-one functions

For the definition of the second concept from the section title, the inverse function, we need to think about the following question: given the output of an unknown input, is it possible to recover the input anyways? Sometimes this can be done, but often it cannot.

Example: Suppose that we have a function planet, that for a given moon of the solar system returns the corresponding planet.If the value of the function is pluto, then we know for sure that the input was charon. If the value was jupiter, then one cannot recover which moon was the input, as jupiter has more than one moon.

A function f is called one-one (one-to-one, 1-1, injective) if for each produced output of f, there is only one corresponding input of f. Formally expressed,

$$\forall x_1, x_2, (f(x_1) = f(x_2) \rightarrow x_1 = x_2).$$

A different expression is

$$\forall y, \forall x_1, x_2, (f(x_1) = y \wedge f(x_2) = y \rightarrow x_1 = x_2).$$

Note the similarity with the formal definition of determinism of functions. The definition is basically a form of determinism "in the other direction", namely from output to input. A function is called many-one (many-to-one) if it is not one-one.

As a side note: The notions of domain of definition and the image of a function f leads to the following true statements.

$$\forall y \in W_f, \exists x \in D_f, f(x) = y.$$

In combination with the property that f is 1-1 we now find that

$$\forall y \in W_f, \exists! x \in D_f, f(x) = y.$$

Examples: The function planet from the previous example is many-one, because in some cases multiple moons belong to the same planet. The function square root, on the contrary, is 1-1: for a given value y, we know that $y \geq 0$. This number is the (positive) square root of $x := y^2$.

Inverse of 1-1 functions

Since functions are special relations, we can also take the inverse of a function. The inverse of a relation R was denoted by R^{-1}. Remember

$x\,R\,y \leftrightarrow y\,R^{-1}\,x.$

When a relation is a function f, the underlying thought is to reverse the process belonging to f. Unfortunately, this is not always possible without loss of determinism, and in general, the inverse of a function is not a function anymore.

We shall denote the inverse of a function f by f^{-1}. In general, f^{-1} is thus a relation. In case it is a function, then the input of f becomes the output of f^{-1} and vice verse. In the usual notation:

$D_{f^{-1}} = W_f, \quad W_{f^{-1}} = D_f.$

Example: The function planet returns the planets of a given moon. The process cannot be reserved without losing determinism. For a given planet it is not always known which moon is considered.

The process for the real-valued function square root can be reversed by the process of squaring: square root^{-1} = square. In this example, we find:

$D_{\text{square}} = W_{\text{square root}} = \{x : x \geq 0\},$
$W_{\text{square}} = D_{\text{square root}} = \{x : x \geq 0\}.$

Example: The function add2 of type $\mathbb{N} \to \mathbb{N}$ has description $\text{add2}(x) := x + 2$. The domain of definition is $\mathbb{N}$, and the image is $\{y : y \geq 2\}$. The process is simple to reverse: given a producible output y of f (i.e., $y \geq 2$), the original input can be obtained by $x := y - 2$. Thus, $(\text{add2})^{-1}$ is the function

$\text{sub2} : \mathbb{N} \to \mathbb{N}, \qquad y \mapsto y - 1,$

with domain of definition $\{y : y \geq 2\}$ and image $\mathbb{N}$.

Theorem 5.3. Let f be a function. The inverse of a function f is a function if and only if f is an 1-1 function.

Proof. First assume that f is 1-1. We show that f^{-1} is a function, i.e.,

$\forall y \forall x_1 \forall x_2, (y\,f^{-1}\,x_1 \wedge y\,f^{-1}\,x_2 \to x_1 = x_2).$

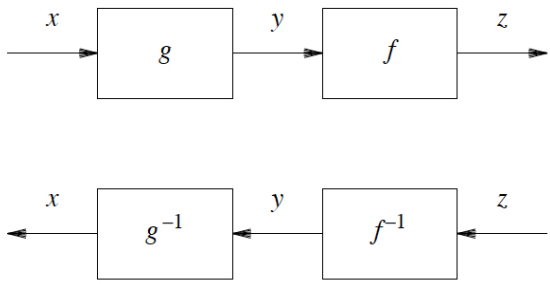
Take y, x_1, x_2 arbitrary and assume that $y\,f^{-1}\,x_1$ and $y\,f^{-1}\,x_2$. Then (by definition of the inverse relation) $x_1\,f\,y$ and $x_2\,f\,y$. Since f is 1-1, it follows that $x_1 = x_2$.

Next, assume dat f^{-1} is a function. We show that f is 1-1, i.e.,

$\forall y \forall x_1 \forall x_2, (y = f(x_1) \wedge y = f(x_2) \to x_1 = x_2).$

Take y, x_1, x_2 in V arbitrary and assume that $y = f(x_1)$ and $y = f(x_2)$. Then (definition of inverse) $y\, f^{-1}\, x_1$ and $y\, f^{-1}\, x_2$. Since f^{-1} is a function, we have $x_1 = x_2$. □

In order to find the inverse of a composition, one take the inverse of each of the components and takes the composition of these in the reverse order. This is illustrated in the following scheme.



Both compositions can be represented as follows:

$$x \overset{f}{\mapsto} y \overset{g}{\mapsto} z, \qquad g \circ f(x) = z.$$
$$x \overset{f^{-1}}{\leftarrow\!\shortmid} y \overset{g^{-1}}{\leftarrow\!\shortmid} z, \qquad f^{-1} \circ g^{-1}(z) = x.$$

This rule allows one to easily compute the inverse of complex function.

Bijective functies

We saw earlier that the inverse of a function f is a function again precisely if f is injective: different input yields different output. Let f be typed as $f : A \to B$. Remember that f is total if the domain of definition is exactly A. The inverse of a total function is (when it is already a function) not automatically a total function of type $B \to A$. Remember that a typed function $f : A \to B$ is surjective is the image is exactly B. One can now prove the following: Let f be a typed function. The inverse of f is a total function if and only if f is injective and surjective. A total function with a total inverse is also called a bijection or a bijective function. A bijection, differently said, is a typed total function that is both injective and surjective.

Take $A :=$ Alphabet $:= \{'a', 'b', \ldots, 'z'\}$ and $B := \{1, 2, \ldots, 26\}$. The typed function $A \to B$ that assign a number to a letter from the alphabet is a bijection. Indeed, different letters have a different number in the alphabet (the function is injective), and each number in B effectively corresponds to a

letter in the alphabet (the function is surjective).

5.5 Counting and cardinality

Counting processes

In Chapter 1 we have studied the number of elements $\#A$ of a set A. This number can be obtained by (often a tedious) direct count of the complete set, or by building up the from easily countable subsets and derive a final answer by a formula, think of the summation formula and the product formula. In this section, we focus ourselves on counting.

Nothing seems to be so easy as counting, but what exactly happens when you count? In classical counting, one calls consecutive numbers: one, two, three, etc., and points to an element of a set with each number. In principle, one creates a function with as input the number and as output the element of the set corresponding to that number. But not every function of that type is a counting function (counting process). There can be three ways in which errors can occur:

- Elements can be counted at least twice.
- Elements are skipped during the counting process.
- In the sequence "one, two, …" at least one number has been skipped.

This corresponds to some nice concepts on functions (injective, surjective, and total) from the previous paragraph.

Remember that a typed function $f : V \to W$ is a bijection if it satisfies the following three conditions:

(i) f is injective, i.e., different elements of V have different f-values in W.

(ii) f is surjective, i.e., each element of W is an f-value of an element of V.

(iii) f is total, i.e., each element of V yields a function value.

Now, one can clearly see the relation between non-injective and counting elements twice. And there is also a relation between non-surjective and skipping an element. And finally the relation between non-total and skipping a number also exists.

A counting process is thus a total bijection between a standard set of numbers and a set that one wants to count. If n is a natural number and

A a set, then the definition of "$\#A = n$" (A has n elements) can now be translated nicely into a concrete statement: there is a total bijection

$$\{1, 2, \ldots, n\} \to A.$$

The elements of the left set are the used numbers for counting. Note that for $n = 0$ the left set is empty!. In that case, the function is also empty (there is no pair with first coordinate in $\emptyset$), and an empty function produces an empty image. Therefore, A is also empty, because otherwise the function is not surjective. And this coincides with $\#A = 0$ having meaning that $A = \emptyset$.

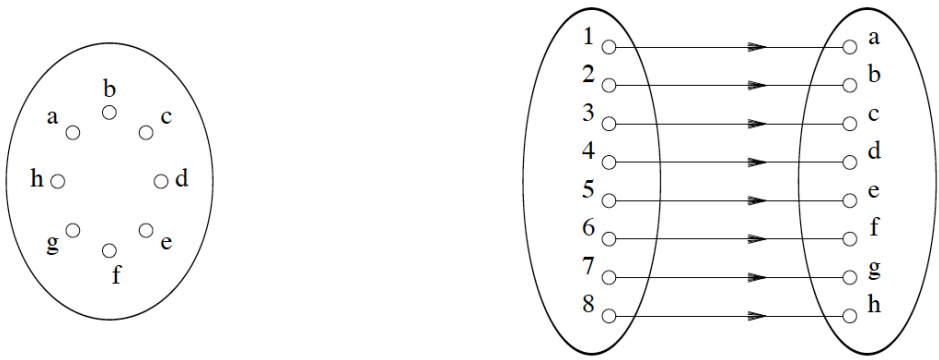


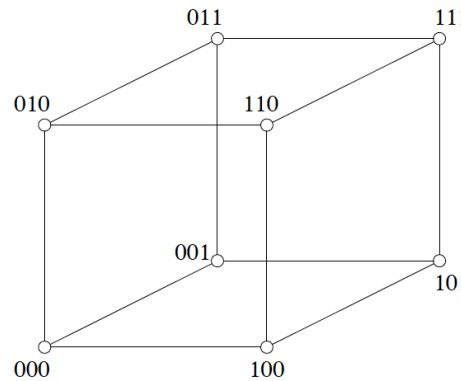
Figure 5.1: Counting a set with eight elements $\{a, b, \ldots, h\}$.

Figure 5.1 illustrates a count $\#A = 8$ via a bijective function $\{1, 2, \ldots, 8\} \to A$. The depicted function is bijective. In the first place, it is a total function (all number $1, \ldots, 8$ are used). Furthermore, we can see that each element of A is counted (f is surjective) and only once (f is injective).

For a simple count, one chooses a number for each element to be counted. For larger examples, one needs to construct a counting function more carefully.

For a given number $n \in \mathbb{N}$, we want to count the corners of the n-cube. Hence, we are dealing with the set $\{0, 1\}^n$. A corner is a sequence of n bits (0 or 1). Such sequences appear, for instance, through the binary representation of numbers, such as it is used internally in computers. The following table represents $0, \ldots, 7$ with three bits. The figure is also a usual representation of a 3-cube.

Number	Binary	Number	Binary
0	000	4	100
1	001	5	101
2	010	6	110
3	011	7	111



This observation allows us to construct a counting function for the corners of the n-cube:

$f : \{0, 1, \ldots, 2^n - 1\} \to \{0, 1\}^n$,

with $f(k) :=$ the binary representation of the number k with n bits. The table above is exactly the function table for $n = 3$. The function is total (each number between 0 (included) and 2^n (not included) has a representation with n bits. The function is injective (different numbers have a different binary representation). And the function is surjective (every row of n bits corresponds to a number between 0 and $2^n - 1$). We conclude that $\#\{0, 1\}^n = 2^n$.

Sets with the same cardinality

We have seen that counting is nothing else than creating a bijection between elements of the set A to be counted and the standard set of number $\{1, \ldots, n\}$ or $\{0, \ldots, n - 1\}$. Such a bijection is called a counting function for A. A counting function yields, apart from a great deal of details, also the conclusion that the two sets are equally large. This observation needs a generalization.

Two sets A and B have the same cardinality if there exists a total bijection $A \to B$. When two sets A and B have the same cardinality, we shall denote that with $A \approx B$. In general (with possibly infinite sets) the expression

"having the same cardinality" is preferable above the expression "are equally large", which might be confusing at some times.

Note that a set has exactly n elements when it has the same cardinality as $\{1,\ldots,n\}$. Hence, it is also clear that two sets with the same number of elements have the same cardinality.

Example: Let V be a set with n points. We show that the power set $\mathcal{P}(V)$ has the same cardinality as the n-cube $\{0,1\}^n$. Remember that a point of the n-cube is an n-pair of bits. In order to construct a bijection of the n-cube with $\mathcal{P}(V)$, we first endow V with an order: a bijection $f : \{1,\ldots,n\} \to V$. Now, each element of V corresponds to one number from $1,2,\ldots,n$. A subset A of V can be unambiguously described by saying which elements of V belong to A. Even better: of each number $1,\ldots,n$ one can tell if the corresponding element in V is an element of A. If a 'yes' is denoted by a '1', and a 'no' by a '0', then we have a bijection

$$g : \{0,1\}^n \to \mathcal{P}(V).$$

The input is a sequence of bits $b :=< b_1,\ldots,b_n >$. The output is a set $A := \{f(i) : b_i = 1\} \subseteq V$. Simply put, the sequence of bits is a sequence of answers to the question "does element $f(i)$ belong to A?" $(i = 1,2,\ldots,n)$.

We have studied the set $V := \{'a','b',\ldots,'g'\}$ earlier, and counted 8 elements ($0 \mapsto 'a', 1 \mapsto 'b'$, etc.). Let A be a subset of V, say $\{'b','d','f'\}$. Then, we can determine an 8-bit word that represents A by 01010100. In this way, we can easily see that the number of subsets of V is exactly $2^8 = 256$. In general, a set with n elements has exactly 2^n subsets. The exponential function for $\#\mathcal{P}(V)$ explains partly the name power set for $\mathcal{P}(V)$.

Example: For finite sets we say that "having the same cardinality" has the same meaning as "has the same number of elements". For infinite sets, the notion of "number of elements" does not have a simple meaning anymore. This gives some counter-intuitive results, which were considered as paradoxes in the past. Here is an illustration.

The sets $\mathbb{N}$ (natural numbers) and $2\mathbb{N}$ (even natural numbers) have the same cardinality. Here is a bijection:

$$f : \mathbb{N} \to 2\mathbb{N}, \qquad f(n) := 2n.$$

Please check this! A real subset can have the same cardinality with the complete set. This is why the statements "equally big" and "has the same number of elements" are naive and need to be avoided. The use of this

terminology has led to considering the infinite as a paradox in the past. When one carefully observes the pattern, one sees already the structure.

- A student that loses half of its student allowance, has acute financial problems.
- A top manager that sees his salary being reduced to half will have to change his lifestyle to accommodate to his new salary.
- An oil multi-millionaire who loses half of his fortune will be disgruntled at most.
- Someone with a fortune having the same cardinality as $\mathbb{N}$ will not experience any difference when we loses half of his fortune.

One observes that how bigger the numbers get, the less is the effect noticeable of loosing half of it. In the infinite the effect is completely gone.

Theorem 5.4. Having the same cardinality is an equivalence relation between sets.

Proof. We need to check three conditions.

- Reflexivity: a set has the same cardinality as itself through the identity function.
- Symmetry: When A has the same cardinality as B, then there exists (according to definition) a bijection $f : A \to B$. The inverser function (!) $f^{-1} : B \to A$ is also a bijection, and this directly shows that B has the same cardinality as A.
- Transitivity: Let A have the same cardinality as B, and let B have the same cardinality as C. Hence, there are bijection $f : A \to B$ and $g : B \to C$. The composite function $g \circ f : A \to C$ is a bijection, and A has the same cardinality as C.

□

The symbol '$\approx$' that we introduced earlier is common for equivalence relations. We finally end with a surprising example of sets having the same cardinality.

Let $\mathbb{N}_0$ be the set of all natural numbers without 0. We construct a bijection

$f : \mathbb{N}_0 \to \mathbb{N} \times \mathbb{N}, \qquad f(x) :=< m, n >$ with $x = 2^m(2n+1)$.

The function process can be described as follows: First find all factors 2 of x, note this number m, and divide these factors out of x. The quotient is an odd number of the form $2^n + 1$. Note this number n.

For instance, 28 is twice divisble by 2, thus $m = 2$. Now, dividing by 2^2 yields $7 = 2 \cdot 3 + 1$, hence $n = 3$. Therefore, $f(28) =< 2,3 >$. Here is a table of function values $f(x)$ for $1 \leq x \leq 9$.

x	1	2	3	4	5
$< m,n >$	$< 0,0 >$	$< 1,0 >$	$< 0,1 >$	$< 2,0 >$	$< 0,2 >$
x	6	7	8	9	
$< m,n >$	$< 1,1 >$	$< 0,3 >$	$< 3,0 >$	$< 0,4 >$	

One can clearly see that

(i) The procedure works for any positive x, i.e., the function f is total.

(ii) Different input numbers x give different pairs, i.e., the function f is injective.

(iii) For each pair $< m,n >$ one can find a suitable number x, i.e., the function is surjective.

Hence, we find that $\mathbb{N}_0 \approx \mathbb{N} \times \mathbb{N}$. Since, $\mathbb{N} \approx \mathbb{N}_0$, we find that $\mathbb{N} \approx \mathbb{N} \times \mathbb{N}$.

We saw that halving/doubling in the infinite does not have an effect. This last example shows that also squaring in the infinite does not have an effect either. In the next paragraph we will see that this has important consequences for classical number systems.

5.6 Some results on cardinality

Finite and countable

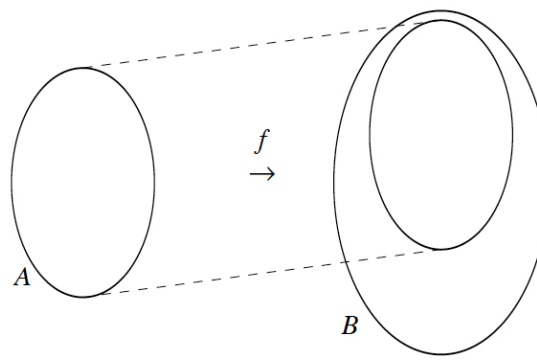
In the previous section we have used bijective functions to check for infinite sets if they had "the same number" of elements. In the positive case, we say that A and B have the same cardinality, and we write this statement as $A \approx B$.

We can give a precise meaning to the intuitive concept of (in)finite sets. A set V is called finite if there exists a number $n \in \mathbb{N}$ with $V \approx \{1, \ldots, n\}$. If such a number does not exist, then the set is called infinite. To a finite set V we can thus associate a quantity n, which is tradionally denoted by $\#V$.

A set V is called countable if it has the same cardinality as $\mathbb{N}$, the set of natural numbers. In the previous section we have seen that $2\mathbb{N}$ and $\mathbb{N} \times \mathbb{N}$ are countable.

Comparison of size
In Theorem 5.4 we have seen that cardinality creates an equivalence relation between sets. The equivalence symbol '$\approx$' suggests something like: equal in size, and it suggests also formulas like $\#A = \#B$. The symbool $\#A$ would then refer to the 'size' of A. One can expect statements that allow one to say that A is bigger or smaller than B, in the style of $\#A \leq \#B$. Here is a proposal:

$\#A \leq \#B \quad :\leftrightarrow \quad$ there is a total injection $f : A \to B$.



It is clear that

(i) (Reflexivity) $\#A \leq \#A$.

(ii) (Transitivity) $\#A \leq \#B \wedge \#B \leq \#C \to \#A \leq \#C$.

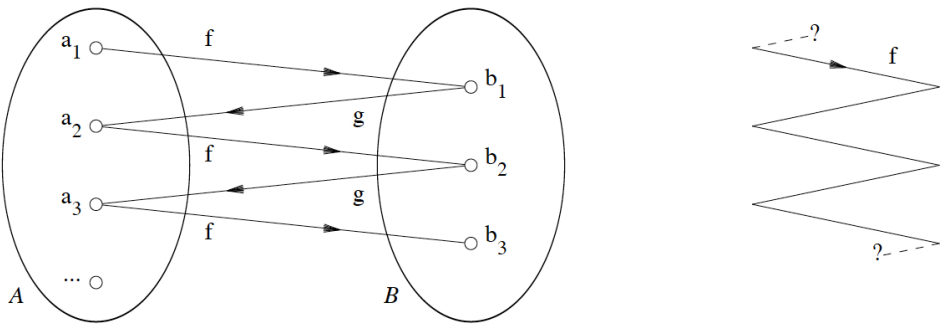
One recognizes two properties of partial orders in here. One only misses one property: anti-symmetry. The usefulness of the definition is partly covered in the following result, that shows anti-symmetry.

Theorem 5.5 (Cantor en Bernstein). Let A and B be two sets. If there are total injective functions $A \to B$ and $B \to A$, then there also exists a total bijection $A \to B$.

In formulas, the theorem says: $\#A \leq \#B \wedge \#B \leq \#A \to \#A = \#B$. This is indeed the anti-symmetry property. The size of sets is (partially) ordered in

this manner. The ordering is even total, but a proof of this result is beyond the scope of this course.

Proof. Let $f : A \to B$ and $g : B \to A$ be total injective functions. Starting with a point $a \in A$, one can create a sequence of points, that lie alternatingly in the sets A and B. If one has point $x \in A$, then the next point is $f(x) \in B$. If one has point $x \in B$, the next point is $g(x) \in A$. The process stops when one encounters a point for the second time (that point has to be point a!).

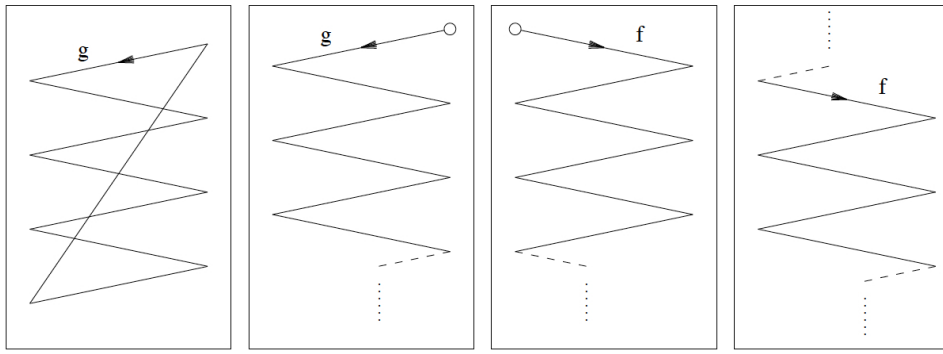


In the left picture, the starting point is $a = a_1$. Starting from this point, we can traverse the process backwards: is there a $x \in B$ with $g(x) = a$? If so, then there is a unique b for the unknown x, and one can ask if there a $x \in A$ with $f(x) = b$, etc. This process only stops when the question is answered with 'no'.

The right graph is a simplification of the sawtooth, produced by a forward and backward process. But it does not reveal how the process ends in both directions. In principle, there are four options (types of sawtooth graphs), and they are depicted below.

Type 1 is created when after some time an earlier used point appears. Type 2 is a sawtooth with a starting point in B and for which the process does not stop. Type 3 is a sawtooth with a starting point in A and for which the process does not stop. Type 4 is the result of a process that never goes back to a point that has been used and never stops either, both for the forward and backward process.

The proof of the theorem now builds upon the following observation: Each point of $A \cup B$ belongs to exactly one sawtooth. Hence, in other words, one has an equivalence relation in $A \cup B$ with the complete sawtooth graphs are equivalence classes.



Type 1 Type 2 Type 3 Type 4

The bijection $A \to B$ that we seek, can now be constructed per equivalence class and the description for points of A in the sawtooth.

Type 1: Use the description $x \mapsto f(x)$.

Type 2: Use the description $x \mapsto g^{-1}(x)$.

Type 3: Use the description $x \mapsto f(x)$.

Type 4: Use the description $x \mapsto f(x)$.

Note that the description yields a point from the B-part of the corresponding sawtooth. The four parts of the description together yields the desired result. □

Application
One can see that for every infinite set V, there is an injection $\mathbb{N} \to V$. Thus, $\#\mathbb{N} \leq \#V$. Indeed, V is not empty (because $\emptyset$ is finite), thus there is an element v_0. But, a one-point set is finite, thus V has another point v_1. A two-point set is finite, thus there is another element v_2, etc. This process yields a list of different points $v_0, v_1, v_2, \ldots$ and with that a total injection $\mathbb{N} \to V$, with $n \mapsto v_n$ $(n \in \mathbb{N})$.

We can conclude that: $\#N$ is the smallest possible 'infinite' quantity.

Application
There is an injection $f : \mathbb{Q} \to \mathbb{N} \times \mathbb{N}$, where a fraction $x := a/b$ $(a, b \in \mathbb{Z}, b \neq 0)$ first is simplified to an equal fraction c/d with $d > 0$ and numbers

c, d that are not divisible by each other. We take $f(x) :=< c, d >$. One the other hand, there is an evident injection $\mathbb{N} \to \mathbb{Q}$ with $f(x) := x$ for each natural number x. Since the set $\mathbb{N}$ is insensitive to squaring, we find that

$$\#(\mathbb{N} \times \mathbb{N}) = \#\mathbb{N} \leq \#\mathbb{Q} \leq \#(\mathbb{N} \times \mathbb{N}).$$

From the theorem of Cantor and Bernstein, it now follows that $\#\mathbb{N} = \#\mathbb{Q}$. Hence,

$$\#\mathbb{N} = \#\mathbb{Z} = \#\mathbb{Q}.$$

The three classic number systems: natural numbers, integers, and fractions all have the same cardinality. Even more precise, they are all countable. The question now remains: how does this relate to the set of real numbers $\mathbb{R}$? For an answer, the following concept is useful.

Sequence in a set
We define a sequence in a set V a total function $f : \mathbb{N} \to V$. The element $f(n)$ is called the n-th element or the n-th term in the sequence. One denotes the sequence f also as $(f(n))_{n=0}^{\infty}$ (this suggests a function table). The index notation is often used as well: $f(n) = a_n$. In fact, a sequence is nothing else than an infinite list.

Example: Sequences of real numbers are perhaps known from calculus. An obvious example can be found in the decimal representation of real numbers: the numbers after the decimal form a sequence in $\{0, 1, \ldots, 9\}$. In this section, we shall mainly use sequences of bits: these are sequences in $\{0, 1\}$. Such a sequence typically has the form:

$$00101010001111010011001001000101000011101111001010010 1\ldots$$

This represents a function $f : \mathbb{N} \to \{0, 1\}$ with $f(0) = 0, f(1) = 0, f(2) = 1, f(3) = 0$, etc.

Function sets
Remember that $\{0, 1\}^n$ (the n-cube) consists of all n-pairs of bits. The set of all (infinite) sequences of bits can (to analogy) be denoted as $\{0, 1\}^{\mathbb{N}}$, or (somewhat more sloppy) as $2^{\mathbb{N}}$. In general, it is customary, for two given sets A and B, to denote the set of all total functions $A \to B$ as B^A.

Example: We have shown that for ($n \in \mathbb{N}$) the n-cube $\{0, 1\}^n$ has the same cardinality as the power set of an n-point set. Something similar appears to hold for infinite-dimensional cubes $\{0, 1\}^V$:

$$\#\{0, 1\}^V = \#\mathcal{P}(V).$$

The proof is almost identical to that of the n-cube: a subset A of V can be coded by a total function $f_A : V \to \{0,1\}$, where $f_A(x) = 1$ if $x \in A$ and $f_A(x) = 0$ otherwise. The function f_A is called the characteristic function of A in V. Reversely, one can construct a subset A of V, for a given function $f : V \to \{0,1\}$, that consists of all $x \in V$ with $f(x) = 1$. In this way, we get a bijection $\{0,1\}^V \to \mathcal{P}(V)$ that ties functions $V \to \{0,1\}$ to subsets of V.

After this intermezzo on function sets we return to the sequences.

The sequence set $\{0,1\}^{\mathbb{N}}$
We compare the size of $\mathbb{N}$ and the one from the interval $[0,1)$ with the size of $\{0,1\}^{\mathbb{N}}$. There is a simple bijection

$$\mathbb{N} \to \{0,1\}^{\mathbb{N}},$$

that assigns to a number $n \in \mathbb{N}$ a sequence of bits that starts with n ones, and has only zeros thereafter. This shows that

$$\#\mathbb{N} \leq \#\{0,1\}^{\mathbb{N}}.$$

On the other hand, every real number can be written in a binary representation instead of a decimal representation. For the numbers between 0 (included) and 1 (not included) this results in a sequence of bits for the numbers after the decimal point. Note that just as $0.4999\ldots = 0.5000\ldots$, we also have in the binary system that (for instance) $0.00111\ldots = 0.0100\ldots$. If we agree on the convention that in case of ambiguity, we choose for the right representation, then we have an injective function $[0,1) \to \{0,1\}^{\mathbb{N}}$.

Reversely, there is an injective function $\{0,1\}^{\mathbb{N}} \to [0,1)$, in which a sequence of bits first is converted to a sequence with a zero added to the every two positions. The longer sequence unambiguously represents a real number in $[0,1)$. This leads to

$$\#\mathbb{R} = \#[0,1) = \#\{0,1\}^{\mathbb{N}}.$$

We now have shown that:

Theorem 5.6. The system of real numbers has the same cardinality as the $\mathbb{N}$-cube.

It seems that in the infinite all 'quantities' are the same. However, the following theorem states that there are indeed more degrees of infinity.

Theorem 5.7 (Cantor). The set $\{0,1\}^{\mathbb{N}}$ does not have the same cardinality of $\mathbb{N}$.

In combination with the earlier observation, we can say that $\#\mathbb{N} < \{0,1\}^{\mathbb{N}}$. We have found a system that is 'even more infinite' than $\mathbb{N}$! More correctly: apparently there are several gradations of infinity.

Proof. The proof of Cantor's Theorem is through the diagonal method. It is a proof by contradiction. Suppose that there is a total bijection $\mathbb{N} \to \{0,1\}^{\mathbb{N}}$. This bijection can be explicitly depicted by the list of bits

row 0:	0	1	0	0	1	...
row 1:	0	1	1	1	0	...
row 2:	1	0	0	1	0	...
row 3:	1	1	1	0	0	...
row 4:	0	0	1	1	1	...

...

Now create a new sequence with the 'complements' of the diagonal:
At position 0 of row 0 there is a '0': note '1'.
At position 1 of row 1 there is a '1': note '0'.
At position 2 of row 2 there is a '0': note '1'.
At position 3 of row 3 there is a '0': note '1'.
etc.

This procedure create a sequence 10110...that cannot be in the list: it differs at position 0 in row 0, at position 1 in row 1, etc. Hence, the counting function is not surjective leading to a contradiction. □

Cantor's Theorem essentially says that the $\mathbb{N}$-cube (and hence the set of real numbers) is not countable.

5.7 Exercises

1. Is the function one-one or many-one? Please provide arguments

   (a) $f : V \to \mathbb{N}$, with V the set $\{x : x \in \mathbb{N}, 10 \leq x < 100\}$ and $f(n)$ the sum of the digits of n (in decimal representation). For example: $f(29) = 2 + 9 = 11$.

   (b) $f : W \to W$, with W the set of all non-empty (alphabet) words, and f a function that converts a word into the word that one gets by shifting each letter one in the alphabet ('z' becomes 'a'). For example, f('this') = 'uijt'.

(c) The function f given by the following table.

x	0	1	2	3	4	5	6	7	8	9	10
$f(x)$	0	1	3	5	9	11	18	17	14	5	2

2. In this exercise, we use the set $A := \{'a', 'b', \ldots, 'z'\}$ of all small uncapitalized letters. Let W_{2+} be the set of all words with at least two letters from A, and let W_2 be the set of all words with exactly two letters from A. We have the function cap : $W_{2+} \to W_2$ that provides a cap of length 2 on each word, so

   cap('tree') = 'tr'.

   We also have the function seqnr : $W_2 \to \{1, 2, \ldots, 1000\}$, where seqnr($x$) is the sequence number of x in the lexicographical sorted list of elements of W_2 up to a maximum of 1000.

   seqnr('af') = 6, seqnr('ba') = 27.

   (a) Give a brief description of the process of the function seqnr ∘ cap.

   (b) Show that the composition seqnr ∘ cap is total.

   (c) Is the function seq ∘ cap surjective? Please explain.

3. In this exercise we have for each $a \in \mathbb{R}$ four functions

   add_a, sub_a, mul_a, div_a $(a \neq 0)$

   of type $\mathbb{R} \to \mathbb{R}$. The description is as follows.

   $\text{add}_a(x) := x + a$;
   $\text{sub}_a(x) := x - a$;
   $\text{mul}_a(x) := x \cdot a$;
   $\text{div}_a(x) := x/a$ $(a \neq 0)$.

   Furthermore, we have disposal of the functions $\exp_2$, root : $\mathbb{R} \to \mathbb{R}$, with descriptions

   $\exp_2(x) := x^2$, root$(x) := \sqrt{x}$.

   (a) Give a description of the function and the domain of definition for the following functions:
   $f := \text{root} \circ \text{add}_1 \circ \text{mul}_2$.
   $f := \text{sub}_3 \circ \exp_2 \circ \text{mul}_5$.

(b) Give the description for the inverse function (assume that we have a function) of the following compositions:
$f := \text{div}_5 \circ \text{add}_2 \circ \text{mul}_3$.
$f := \exp_2 \circ \text{add}_1$.

(c) Rewrite the function $f : \mathbb{R} \to \mathbb{R}$ with $f(x) := (3\sqrt{x}+1)^2$ in parts of the above defined functions.

4. Consider the function

$$f : \mathbb{R}^2 \to \mathbb{R}, \quad f < x_1, x_2 >:= \sqrt{x_1^2 + x_2^2}.$$

Note that based on the classical Pythagorean theorem on rectangular triangles, this is the distance between a point $< x_1, x_2 >$ to the origin $< 0, 0 >$.

(a) Describe the corresponding equivalence relation and its equivalence classes.

(b) Describe (as simple as possible) the complete system of representatives.

5. Equivalence relations from functions.

(a) Let the function $f : \mathbb{N}^2 \to \mathbb{Z}$ be given by the description $f < a, b >:= a - b$. Show that the corresponding equivalence relation in $\mathbb{N} \times \mathbb{N}$ is exactly the relation in Exercise 3(b) from Chapter 4.

(b) Let the function $g : \mathbb{Z} \times (\mathbb{Z} \setminus \{0\}) \to \mathbb{Q}$ be defined by the description $g < a, b >:= a/b$ (remember that $\mathbb{Q}$ is the set of fractions). Show that the corresponding equivalence relation in $\mathbb{Z} \times (\mathbb{Z} \setminus \{0\})$ is exactly the relation of exercise 3(c) in Chapter 4.

(c) Let $m \geq 2$ be an integer and let $h : \mathbb{Z}to\mathbb{N}$ be the function that assigns to a number z its remainder when z is divided by m. For example, for $m = 4$, we have $f(17) = 1$, since $17 = 4 \cdot 4 + 1$, and $h(-17) = 3$, since $-17 = -5 \cdot 4 + 3$. Show that the corresponding equivalence relation is exactly the relation "congruence modulo m".

6. Equivalence relations and functions. Of a set Students, all the members are checked for their capacities. The test results can be represented as a total function

$$\text{capacity} : \text{Students} \to \text{Scores},$$

where Scores := {0, 1, 2, 3, 4, 5} has the natural ordering $0 < 1 < \cdots < 5$. The function capacity yields a number (score) for any student as input. The higher the score, the more capable the person is. Let R be the binary relation in Students with description

$$x\,R\,y :\leftrightarrow \text{capacity}(x) \leq \text{capacity}(y).$$

(a) Show that the relation R is reflexive and transitive.

(b) Suppose that Students := $\{p_1, \ldots, p_9\}$, and let the function capacity be given by the following table.

person:	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9
capacity:	3	0	5	4	2	5	3	4	1

Explain why the relation R is not an order relation in Students.

(c) (Assuming the general case again). Let S be the inverse relation of R. Give a description of S using the function capacity, and show that the relation $R \cap S$ is an equivalence relation in the set Students.

(d) Give in the concrete situation of (b) the equivalence class of person p_4 with the equivalence relation of (c).

5.8 Background

The notion of a function is, next to the notion of a set, the most fundamental in mathematics. The word "function" was first used in contemporary mathematics by Gottfried Wilhelm von Leibniz (1646–1716) in 1692. The $f(x)$-notation comes from Alexis Clairaut (1719–1765). The concept of functions was formulated in 1837 by Peter Gustav Lejeune-Dirichlet (1805–1859), and has slowly developed from calculus-like formulas with the usual operations addition, subtraction, multiplication, division, powers, and roots. The informal description of functions as a relation between inputs and outputs with omitting the process details can also be found in reference [2].

There are various programming languages based on the idea of functions. This style of programming is better known as functional programming. The oldest and most well-known language of this type is LISP (John, McCarty, 1960). For an introduction into the modern functional language Miranda, see [4].

Cardinality of sets dates back to the end of the nineteenth century based on ideas of Georg Cantor. Some results and methods of proofs in this the-

ory have been of influence on later subjects such as Turing machines and completeness.

Chapter 6

Induction and Recursion

Goals of this chapter:

- Knowledge of arithmetic rules in $\mathbb{Z}$ with respect to addition, multiplication, and the natural ordering.
- Give and understand a proof by induction.
- Knows the difference between normal and strong induction.
- Knows the relation between induction and the well-ordering principle in $\mathbb{Z}$.
- Calculate terms in a recursively defined sequence.
- Check a given formula for a recursively defined sequence by proof of induction.
- Describe the transitive closure of a relation.

6.1 The integers

In this course, we have referred quite often to the set $\mathbb{Z}$ of integers, and the usual operations addition '$+$' and multiplication '$\cdot$'. This chapter is about methods of proof that a specific for integers. This paragraph provides an overview of the fundamental rules for arithmetic in $\mathbb{Z}$.

When carrying out arithmetic operations, the following priority rule is in force: multiplication has higher priority over addition. For instance, the expression $x \cdot y + z$ with parentheses has the meaning $(x \cdot y) + z$. If this is

not the preferred order, then parentheses are used: $x \cdot (y + z)$. Expressions with operators, such as $(a + x) \cdot (b + y)$ are called terms.

Equality in $\mathbb{Z}$ is a reflexive relation: $x = x$ for all x. Furthermore, there is a general principle from logic: in a formula one can replace equal parts by other equal parts (the substitution rule). More precisely, when $t_1 = t_2$ (for two given terms t_1 and t_2), then a formula is equivalent with a formula in which t_1 is replaced by t_2.

Addition
The set $\mathbb{Z}$ is equipped with a binary operation '+', called addition (sum), and for which the following rules hold:

Rules for addition	
(S1)	Addition is associative: $\forall x \forall y \forall z, (x + y) + z = x + (y + z)$.
(S2)	Addition is commutative: $\forall x \forall y, x + y = y + x$.
(S3)	Neutral element for addition 0: $\forall x, x + 0 = 0 + x = x$.
(S4)	Inverse: for every x there is in inverse element $-x$ such that $\forall x, x + (-x) = (-x) + x = 0$.

The notation $x - y$ is an abbreviation for $x + (-y)$. Instead of a 'neutral element for addition' one also speaks of a zero element.

Multiplication
The set $\mathbb{Z}$ is also equipped with the binary operation '$\cdot$', called multiplication (product), for which the following rules hold.

Rules for multiplication	
(V1)	Multiplication is associative: $\forall x \forall y \forall z, (x \cdot y) \cdot z = x \cdot (y \cdot z)$.
(V2)	Multiplication is commutative: $\forall x \forall y, x \cdot y = y \cdot x$.
(V3)	Neutral element for multiplication 1: $\forall x, x \cdot 1 = 1 \cdot x = x$.

Instead of a 'neutral element for multiplication' one also speaks of a unit element.

Distributive law and zero-divisors

The combination of addition and multiplication and the corresponding neutral elements have to further satisfy the following rules.

1. (N) Law of zero-divisors: $\forall x \forall y, (x \cdot y = 0 \rightarrow x = 0 \vee y = 0)$.
2. (D) Distributive laws: $\forall x \forall y \forall z, x \cdot (y + z) = (x \cdot y) + (x \cdot z)$
   $\forall x \forall y \forall z, (x + y) \cdot z = (x \cdot z) + (y \cdot z)$.

Ordering
In the set $\mathbb{Z}$ there is a total ordering $\leq$ available. This assumes the following laws.

Total ordering

(O1) Reflexive: $\forall x, x \leq x$.

(O2) Anti-symmetry: $\forall x \forall y, (x \leq y \wedge y \leq x \rightarrow x = y)$.

(O3) Transitive: $\forall x \forall y \forall z, (x \leq y \wedge y \leq z \rightarrow x \leq z)$.

(O4) Total: $\forall x \forall y, (x \leq y \vee y \leq x)$.

The definition of natural numbers can be given as:

$\mathbb{N} := \{x : x \in Z, 0 \leq x\}$.

For the notion of (total) ordering in general sets, one can read Chapter 3. In this chapter one can also find information on the strict ordering '$<$' that can be obtained as:

$x < y \quad : \quad x \leq y \wedge x \neq y$.

Finally, the following laws with respect to the ordering and the operations addition and multiplication.

The laws, that we have mentioned so far, are not characteristic for integers. The fractions and the real numbers satisfy all the above mentioned laws as well. Arithmetic with a modulus satisfy all laws that are not related to orderings, with a possible exception of (N). With these laws, one can show all laws that are needed for arithmetic with the usual operations and the natural orderings.

(0&1) Inequality of 0 and 1: $0 < 1$.

(O+) Stability for addition: $\forall x \forall y \forall z, (x \leq y \rightarrow x + z \leq y + z)$.

(O·) Stability for multiplication (positive numbers): $\forall x \forall y \forall z, (x \leq y \wedge 0 \leq z \rightarrow x \cdot z \leq y \cdot z)$.

6.2 Induction

The set $\mathbb{N} := \{0, 1, 2, 3, \ldots\}$ of all natural numbers is of fundamental importance for both mathematics and for applications outside mathematics. It is therefore not surprising that for statement about natural numbers (and with extension: the integers), special techniques of proof have been developed.

In this section, we will focus on a method of proof that is quite specific for a universe such as $\mathbb{N}$ or $\mathbb{Z}$, namely, the principle of induction. We will also discuss a method of construction for sequences (recursion) that is very convenient for proving by induction.

We first start with some examples on natural numbers.

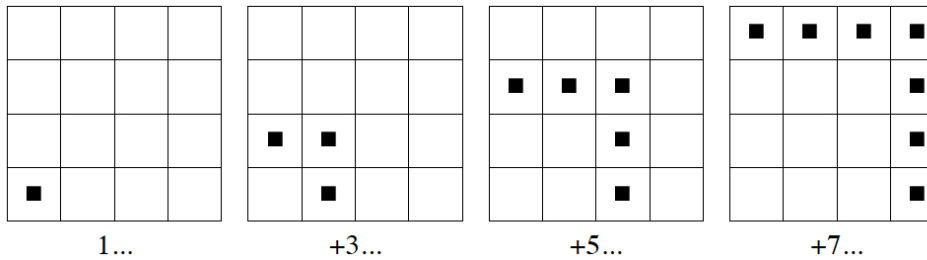
Example 1
For every natural number n, the sum of the first n number is equal to $\frac{1}{2}n(n+1)$. Thus, $0 + 1 + 2 + \cdots + n = \frac{1}{2}n(n+1)$. This is the so-called summation formula of Gauss. The summation in the left part can be represented by the $\sum$-sign.

$$\sum_{k=0}^{n} k, \qquad \text{or, without the zero ,} \qquad \sum_{k=1}^{n} k.$$

Example 2
For every natural number $n \geq 1$, the sum of the first n odd numbers is equal to n^2. Thus, $1 + 3 + \cdots + 2n - 1 = n^2$. The summation in the left part can be represented as

$$\sum_{k=1}^{n} (2k - 1).$$

The geometric intuition behind the equality is depicted in the following figure.



Example 3
A diagonal in a polygon is a line between two not-consecutive corners. The line passes through the figure internally, so to speak. Let $n \geq 3$ a natural number. Then, the number of diagonals in a n-polygon is exactly $\frac{1}{2}n(n-3)$.

Example 4
Given is a natural number $d > 0$. Every natural number n has a quotient q and a remainder r after division by d:

$$\exists q \in \mathbb{N}, \exists r \in \mathbb{N}, (0 \leq q \wedge 0 \leq r < d \wedge n = d \cdot q + r).$$

This is (apart from some details) the so-called Euclid's algorithm for division, which says that when a number n is divided by d, there is a unique pair of number q (= quotient) and r (= remainder) such as given in the formula.

The principle of full induction
In each of the example lists before, one has to deal with statements of the form $\forall n \in \mathbb{N}, Pn$. It is known that each statement is true, but we cannot use the methods of proof of the previous chapters. With a "for all" statement in the set of natural numbers, there is an elegant alternative, which is known as the principle of full induction. The strategy to follow is the following.

	Premisses	Target:
Starting argument	[arithmetic rules]	$\forall n \in \mathbb{N}, P(n)$
Working argument 1	[arithmetic rules]	$P(0)$
Working argument 2	[arithmetic rules]	$\forall n \in \mathbb{N}, (P(n) \rightarrow P(n+1))$

The rules of arithmetic refer to the laws for integers, described in the previous section. The goal in the second working argument is a "for-all implication". Such an argument can be further simplified with the usual combination strategy. This leads to the following strategy.

Full induction	Premisses	Target:
Starting argument	[arithmetic rules]	$\forall n \in \mathbb{N}, P(n)$
Working argument 1	[arithmetic rules]	$P(0)$
Working argument 2	[arithmetic rules], $n \in \mathbb{N}, P(n)$	$P(n+1))$

The proof is thus split into two parts:

(i) Proof $P(0)$. This argument is called the basis of induction.

(ii) Show that $P(n+1)$ follows from $P(n)$ for arbitrary $n \in \mathbb{N}$. This fase is called the induction step of n to $n+1$, and $P(n)$ is the induction hypothesis.

The conclusion is that $P(n)$ is true for all $n \in N$. With this strategy, one can also see the validity of the following form of argumentation: (universe $\mathbb{N}$)

$$\begin{cases} \text{arithmetic rules}, P(0) \\ \forall n \in \mathbb{N}, (P(n) \rightarrow P(n+1)) \\ \therefore \forall n \in \mathbb{N}, P(n). \end{cases}$$

Example 1: the summation formula of Gauss
Theorem: For all $n \geq 0$ we have $0+1+2+\cdots+n = \frac{1}{2}n(n+1)$.

The formula $0+1+2+\cdots+n = \frac{1}{2}n(n+1)$ is the sentence $P(n)$ with subject n. This sentence is shown to be true with induction on $n \geq 0$.

Proof:
Basis: $n = 0$. The left and the right part of the formula are both equal to 0.
Step $n \mapsto n+1$: Let $n \geq 0$ arbitrary and assume that the formula is valid for n (induction hypothesis). We show that the formula for $n+1$ holds.

$$\begin{aligned} 0+1+2+\cdots+n+(n+1) &= (0+1+2+\cdots+n)+(n+1) \\ &\stackrel{!}{=} \frac{1}{2}n(n+1)+(n+1) \quad \text{(induction hypothesis)} \\ &= \frac{1}{2}(n+2)(n+1), \end{aligned}$$

from which $P(n+1)$ follows. Note that we place an exclamation mark above the equal sign where the induction hypothesis is used.

Induction from a starting number
In some cases , the basis of induction is not 0, but some integer b and the variable n assumes values in $\{x : x \in \mathbb{Z}, x \geq b$. In Example 2 we discussed a summation formula that is valid for all number $n \geq 1$. In this case, we have a starting number $b = 1$. In Example 3 (on diagonals in a polygon) we have a formula valid for all $n \geq 3$. In this case, we have a starting number $b = 3$.

In such situation, we speak of induction on $n \geq b$ or of induction on n with basis b. The adjustment to the strategy for this more general case is obvious:

	Premisses	Target:
Starting argument	[arithmetic rules]	$\forall n \geq b, P(n)$
Working argument 1	[arithmetic rules]	$P(b)$
Working argument 2	[arithmetic rules], $n \geq b, P(n)$	$P(n+1))$

Strong induction
There is an induction strategy with a more extensive induction-hypothesis and (in theory) without basis:

(i) Take $n \geq b$ completely arbitrary.

(ii) Assume that $P(k)$ is valid for all integers k with $b \leq k < n$ and prove that $P(n)$ holds.

The conclusion is that $P(n)$ is valid for all $n \geq b$. At first, the strategy has the following form:

	Premisses	Target:
Starting argument	[arithmetic rules]	$\forall n \geq b, P(n)$
Working argument	[arithmetic rules]	$\forall n \geq b, (\forall k, (b \leq k < n \to P(k)) \to P(n))$

The working argument has a “for-all implication” again as target. The usual combination strategy leads to the following result.

Strong induction	Premisses	Target:
Starting argument	[arithmetic rules]	$\forall n \geq b, P(n)$
Working argument	[arithmetic rules], $n \geq b$	
	$\forall k, (b \leq k < n \to P(k))$	$P(n)$

This variant of the induction principle is also called the strong induction principle. With the strong principle of induction, one can directly see that the following line of argumentation is valid (universe $\mathbb{Z}$):

$$\begin{cases} \text{arithmetic rules,} \\ \forall n \geq b, ((\forall k, (b \leq k < n \rightarrow P(k))) \rightarrow P(n)), \\ \therefore \forall n \geq b, P(n). \end{cases}$$

Please note the induction hypothesis in case $n = b$: since there is no k for which $b \leq k < b(= n)$, one needs to show $P(b)$ without any hypotheses. As illustration of strong induction we discuss Example 4 more extensively.

Example
Theorem: Let $d \in \mathbb{N}$ with $d > 0$. There exist for every $n \geq 0$ two numbers $q \in \mathbb{N}$ and $r \in N$, such that $n = d \cdot q + r$ and $r < d$.

The formula $\exists q \in \mathbb{N} \exists r \in \mathbb{N}, (n = d \cdot q + r \wedge r < d)$ (with free variable n) is taken as sentence $P(n)$ with subject n. We show that validity of this sentence with strong induction on $n \geq 0$.

Proof: Let $n \geq 0$ be arbitrary and assume (induction hypothesis) that for all k with $0 \leq k < n$ the quotient and the remainder exist when dividing k by d. There are two cases: $n < d$ and $n \geq d$.

In the first case $n < d$, we have a quotient $q = 0$ and a remainder $r = n$.

In the second case $n \geq d$, take $n' = n - d$. Then, $0 \leq n' < n$ and by induction-hypothesis there exists a quotient q' and remainder r' after dividing n' by d:

$n' = d \cdot q' + r', \qquad r' < d.$

With $q := q' + 1$ and $r' = r$, we find that

$n = n' + d = (d \cdot q' + r') + d = d \cdot (q' + 1) + r' = d \cdot q + r.$

Although the principle of strong induction has no explicit provision for a 'basis', in practice, one sees one or more basis situations that are proven separately from the real proof. In the previous example, we have studied the case $n < d$ separately.

6.3 The well-ordering principle in $\mathbb{Z}$

The natural ordering in $\mathbb{Z}$
Remember the natural ordering $\leq$ in the set $\mathbb{Z}$ of all integers. One of the

properties is that it is a total ordering: each two elements x and y are comparable in the sense that $x \leq y$ or $y \leq x$. This ordering has another property, related to the principle of induction. We first formulate some concepts that will help us.

Bounded sets
Let V be a set, $\leq$ a partial order in V, $A \subseteq V$, and $b \in V$. Then, b is called a lower bound for A when $\forall a \in A, b \leq a$. We called b an upper bound for A when $\forall a \in A, a \leq b$. The set A is called bounded from below when A has a lower bound. The set A is called bounded from above, when A has an upper bound.

Example: In $\mathbb{Z}$ with the natural ordering, the set $\{x : x > 0\}$ has a lower bound but not an upper bound. The set of all even numbers in $\mathbb{Z}$ has no lower nor an upper bound.

The well-ordering principle
The envision property of the natural ordering in $\mathbb{Z}$ can now be formulated as follows.

> "every non-empty set bounded from below in $\mathbb{Z}$ has a smallest element".

This is called the well-ordering principle. Naturally, the analogous properties that a non-empty set bounded from above in $\mathbb{Z}$ has a largest element. In fact, the first principle implies the other. We will not use this version of the principle in the sequel.

Example: The cited property is specific to $\mathbb{Z}$ and does not hold in most other orderings. The set $\{x : x > 0\}$ in $\mathbb{R}$ is not empty and bounded from below (by 0, for instance). However, it does not have a smallest element.

Theorem 6.1. Given the well-ordering principle, the principle of induction holds.

Proof. Let P be a predicate, $b \in \mathbb{N}$, and assume that
(i) $P(b)$
(ii) $\forall n \geq b, (P(n) \rightarrow P(n+1))$.
Assume that the well-ordering principle holds, then we shall show that $\forall n \geq b, P(n)$ using proof by contradiction. Assume that $P(n)$ does not hold for all $n \geq b$. Then the set

$$V := \{x : x \in \mathbb{Z}, x \geq b, \neg P(x)\}$$

is not empty. Moreover, the set V is bounded from below (by b for example). According to the well-ordering principle, V has a smallest element. Let n be this element. Then, $n \geq b$ and $\neg P(n)$. Because, $P(b)$ is true according to (i), we have $n \neq b$. Thus, $n > b$ and $n - 1 \geq b$. Moreover, $n - 1 \notin V$, since n is the smallest element of V. Therefore, $P(n-1)$ and $\neg P(n)$ are in contradiction with (ii). □

Theorem 6.2. If the principle of induction holds, then the well-ordering principle holds.

Proof. Let $V \subseteq \mathbb{Z}$ a set bounded from below and non-empty. Suppose that the principle of induction holds, then we will show by contradiction that V has a smallest element. Assume that V does not have a smallest element. Let b be a lower bound for V and use the following sentence:

$P(n) := b \leq n \wedge \forall k \in \mathbb{Z}, (k < n \rightarrow k \notin V)$.

We apply the principle of induction to show that $\forall n \geq b, P(n)$.

Basis $n = b$: Since b is a lower bound of V, we have: if $k \in V$ dan $b \leq k$, or put differently, if $k < b$ then $k \notin V$. Consequently, $P(b)$ is valid.

Step $n \mapsto n + 1$. Take an arbitrary $n \geq b$ and assume that $P(n)$ does not hold, i.e., $b \leq n$ and $k < n \rightarrow k \notin V$ for each $k \in \mathbb{Z}$. Notice that $n \notin V$, if not, n should have been the smallest element of V. Then, $b \leq n + 1$ and if $k < n + 1$ then either $k < n$ and thus $k \notin V$ (hypothesis), or $k = n$ and thus also $k \notin V$. This shows that $P(n + 1)$ holds.

Due to the principle of induction, we can now conclude $\forall n \geq b, P(n)$. Verbally, for each $n \geq b$ and each $k < n$ we have $k \notin V$. But this means that V is empty! □

6.4 Sequences and recursion

A sequence in a set

Remember that a sequence in a set V is a total function $f : \mathbb{N} \rightarrow V$. The element $f(n)$ is called the n-th element or the n-th term of the sequence. One usually denotes the sequence f also as $(f(n))_{n=0}^{\infty}$ (this suggest a function table). Often the index notation is used as in $f(n) = a_n$.

The domain of definition of a sequence does not need to be $\mathbb{N}$ at all times, but can be more general of the form $\{x : x \in \mathbb{Z}, x \geq b\}$, where b is the starting index (usually 0 or 1).

Example: an example of a sequence in $\mathbb{N}$ is: 2, 4, 6, 8, ….. This suggests a formal definition of a sequence with the use of a function $f : \{1, 2, 3, \ldots\} \to \mathbb{N}$, where the n-th term is given by $f(n) := 2n$.

Example: an example of a sequence in $\mathbb{R}$ is: $1, 1/2, 1/3, 1/4, 1/5, \ldots$. This suggest a formal definition of a sequence with the use of a function $f : \{1, 2, 3, \ldots\} \to \mathbb{R}$, where the n-th term is given by $f(n) := 1/n$.

Recursively defined sequences
Suppose that we want to have a description for a sequence f in a set V. This can be done in a direct manner as in the previous examples. However, often times sequences are defined recursively. One needs to envision that the n-th term is described by the terms before the n-th. This is called a resursive definition. Here are some examples.

Example 1: Arithmetic sequence
An arithmetic sequence can be described by a starting term c and a difference d with the following recursive definition:

$$f : \mathbb{N} \to \mathbb{R}, \quad f(0) := c, \quad f(n+1) := f(n) + d.$$

The sequence thus starts as $c, c + d, c + 2 \cdot d, \ldots$. It is clearly to see with induction that the n-th term of this sequence can be described directly by the function $f(n) = c + n \cdot d$.

Example 2: Geometric sequence
A geometric sequence can be described by a starting term c and a ratio r with the following recursive definition:

$$f : \mathbb{N} \to \mathbb{R}, \quad f(0) := c, \quad f(n+1) := f(n) \cdot r.$$

The sequence thus starts as $c, c \cdot r, c \cdot r^2, \ldots$. It is clearly to see with induction that the n-th term of this sequence can be described directly by the function $f(n) = c \cdot r^n$.

Example 3: Factorial
The factorial $f(n)$ is a number n that is defined as follows

$$f : \{x : x \geq 1\} \to \mathbb{N}, \quad f(1) := 1, \quad f(n+1) := (n+1) \cdot f(n).$$

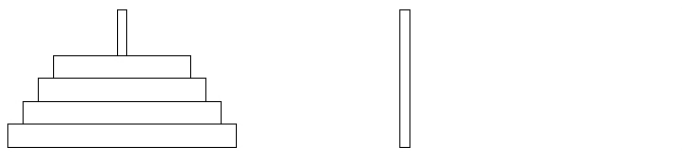
The usual notation is $n!$ (read as n factorial). A direct description is hard to give and usually is given in a suggestive way $1 \cdot 2 \cdot \cdots \cdot n$, which is not convenient in proofs. The recursive definition, on the contrary, works much better.

Example 4: The Towers of Hanoi
There are $n \geq 1$ perforated coins of different diameter, that are attached to a stick, the largest below. There are two other sticks. The aim is to transfer the pile of coins from one stick to the other, with the third stick for help. There are only two rules that need to be satisfied at all times.

- Only one coin can be transferred at a time.
- A largest coin cannot be on top of a smaller one.

Define the sequence f by the following description

> $f(n)$ is the number of transfers that one needs at minimum to reach the end goal with n coins ($n \geq 1$).



A simple argument shows that

$$f(1) := 1, \quad f(n+1) := 2 \cdot f(n) + 1.$$

This defines a sequence recursively. A direct description for the n-th term of the sequence can be given, but requires some proof.

Example 5: Exponents of a relation
Let R be a binary relation in a given set V. Remember that such relations are a subset of $V \times V$, and that the collection of all such subsets are denoted by $\mathcal{P}(V \times V)$. We now give a recursive description for the n-fold composition of R as follows (for $n \geq 1$)

$$f : \mathbb{N}_0 \to \mathcal{P}(V \times V), \quad f(1) = R, \quad f(n+1) = f(n) \circ R.$$

We calculate a few terms

$$f(2) = R \circ R,$$
$$f(3) = f(2) \circ R = R \circ R \circ R,$$
$$f(4) = f(3) \circ R = R \circ R \circ R \circ R,$$
etc.

This suggests a better notation: instead of $f(n)$ we write R^n (the n-th power of R). The definition can be represented more conveniently by

$$R^1 := R, \quad R^{n+1} := R^n \circ R.$$

This construction yields a sequence in $\mathcal{P}(V \times V)$ (!) that consists of all 'repeated' compositions of R with itself. One could also have used the recursive definition $R^{n+1} := R \circ R^n$. This gives the same result as the first definition.

Proofs by induction are frequently used in sequences that are recursively defined. As application we discuss how the value of a numeric function, described recursively, can be determined more directly. In general, this is a difficult problem for which special techniques exist. When a 'direct' formula is proposed, however, then a proof by induction can be used to prove the correctness of the formula.

Example: the Towers of Hanoi
Statement: $f(n) = 2^n - 1$.
Proof: with induction on $n \geq 1$.
Basis $n = 1$: $f(1) = 1$ and $2^1 - 1 = 1$.
Step $n \mapsto n+1$: Assume that $f(n) = 2^n - 1$ (induction hypothesis). Then

$$f(n+1) = 2f(n) + 1 \stackrel{!}{=} 2(2^n - 1) + 1 = 2^{n+1} - 1,$$

where the second equality uses the induction hypothesis.

Transitive closure
Let R be a relation in a set V. The transitive closure is the union of all relation R^n for $n \geq 1$. The explicit definition for the transitive closure of R is:

$$\exists n \geq 1, < x, y > \in R^n.$$

An alternative description is as follows. An R-path in the set V is a finite list of points

$$< x_0, x_1, \ldots, x_n >$$

where consecutive point are in relation

$$x_0 \, R \, x_1, \quad x_1 \, R \, x_2, \quad \ldots \quad , x_{n-1} \, R \, x_n.$$

The transitive closure of R consists of exactly those pairs $< x, y >$ for which an R-path in V exists that starts with x and ends with y (an R-path from x to y).

A Hasse diagram is a graphical representation of a relation, taken from a partial order. One condition is that each pair of the order relation must be from a 'path of arrows' in the Hasse diagram. From this view, the partial order relation is the transitive closure of the Hasse diagram.

Theorem 6.3. Let R be a relation in a set. The transitive closure of R is the smallest transitive relation that contains R.

This theorem is quite compact in its formulation, but it contains multiple statements.

(i) The transitive closure R_{trans} of R is indeed a transitive relation.

(ii) R_{trans} contains R, i.e., $R \subseteq R_{\text{trans}}$.

(iii) If $R \subseteq S$, where S is a transitive relation, then $R_{\text{trans}} \subseteq S$.

6.5 Exercises

1. Prove the following statement with induction on n.
   (a) $n^2 \leq 2^n$, $(n \geq 4)$.
   (b) 7 is a divisor of $3^{2n+1} + 2^{n-1}$, $(n \geq 1)$.
   (c) $\frac{1}{1\cdot 2} + \frac{1}{2\cdot 3} + \cdots + \frac{1}{n\cdot(n+1)} = \frac{n}{n+1}$, $(n \geq 1)$. Note that the left part can be written as $\sum_{k=1}^{n} \frac{1}{k\cdot(k+1)}$.
   (d) $\frac{1}{1\cdot 3} + \frac{1}{3\cdot 5} + \frac{1}{5\cdot 7} + \cdots + \frac{1}{(2n-1)\cdot(2n+1)} = \frac{n}{2n+1}$, $(n \geq 1)$. Note that the left part can be written as $\sum_{k=1}^{n} \frac{1}{(2k-1)\cdot(2k+1)}$.

2. (a) A sequence of real numbers $(t_n)_{n=1}^{\infty}$ is defined recursively as follows
   $$t_1 := 2, \quad t_{n+1} = 2t_n - 2n + 3, \quad (n \geq 1).$$
   Write down the first five terms of the sequence $(t_n)_{n=1}^{\infty}$.
   (b) Prove by induction that $t_n = 2^{n-1} + 2n - 1$ for all $n \geq 1$.

3. A set with n elements has exactly 2^n subsets. Prove this with induction on $n \geq 0$.

4. When V is a set with n elements, there are exactly $n!$ total bijections $\{1, \ldots, n\} \to V$. Prove this with induction on $n \geq 1$.

5. Determine the transitive closure of the following relation.
   (a) The relation R in $\mathbb{N}$ with description "is a direct predecessor of". Explicitly: $x \, R \, y :\leftrightarrow x + 1 = y$.

(b) The relation R in the set of all not-empty words with '0' and '1', for which the following description holds. A word w followed by '0' is related to a word w followed by '10' and to a word w followed by '1'. Informally, if a sequence of bits ends with a '0', then the last bit can be replaced by a '10' or a '1'. For example

$$0\,R\,10, \qquad 1110\,R\,1111.$$

6.6 Background

Proofs by inductions belong to the mathematical arsenal for more than 400 years. The usual name "Mathematical Induction" for the principle of induction was coined by Augustus DeMorgan (1806–1871) in 1838. An important modern application of induction related to checking the correctness of 'while-loops' in computer programs. A while-loop is an instruction that is repeated until a certain condition is or is not satisfied. To be sure that the loop ends with a specific end result, one usually applied proofs by induction.

An early example of a recursively defined sequence comes from Leonardo of Pisa (1175–1250), better known as Fibonacci (= son of Bonaccio). This mathematician described a sequence of numbers, with which he could model the growth of a population of rabbits under ideal circumstances.

Recursive definitions are usually used to construct special sets (such as the transitive closure of a set). For example, one can see an artificial language as a collection of statements. This set is usually recursively defined through grammatical 'production rules', see [5].

The process of the transitive closure is in a certain way the opposite of the process that corresponds to the design of the Hasse diagram for partial orders. The 'thinned' information in a Hasse diagram needs to be extended with information on the ordering. For calculating the transitive closure, the efficient Warshall algorithm exists [10].

Bibliography

[1] K.P. Bogart. Discrete Mathematics. Health & Co., 1988.

[2] D.J. Booth. Foundations of Discrete Mathematics for Computing. International Thomson Computer Press, London, 1995.

[3] D. van Dalen, K. Doets, and H.C.M. de Swart. Verzamelingen, naïef, axiomatisch, en toegepast. Oosterhoek, Scheltema & Holkema, 1995.

[4] S. Epp. Discrete Mathematics with Applications. Thomson Brooks/Cole Publ. Co., Belmont USA, 3d edition edition, 2004.

[5] W.K. Grassmann and J.-P. Tremblay. Logic and Discrete Mathematics. Prentice Hall, New Jersey, 1996.

[6] P.R. Halmos. Naïeve verzamelingenleer. Aula pockets #372, Het spectrum, 1968.

[7] R. Johnsonbaugh. Discrete Mathematics. Prentice Hall, Upper Saddle River, New Jersey, 2001.

[8] P.C. Jorgensen. Software Testing: a craftsman's approach. CRC Press, 2002.

[9] N. Nissanke. Introductory logic and sets for computer scientists. Addison Wesley Longman, 1999.

[10] K.H. Rosen. Discrete Mathematics and Its Applications. McGraw Hill International Editions, Singapore, 1999.

[11] G. Schmidt and Th. Ströhlein. Relations and Graphs. EATCS monographs on Theoretical Computer Science. Springer Verlag, Berlin, 1993.

[12] J.K. Truss. Discrete Mathematics for Computer Science. Addison-Wesley, 1999.