Discrete Mathematics and Logic

Autumn Semester 2012

G. Jäger

Assistance and Support:

S. Eberhard and R. Zumbrunnen

Contents

Chapter 1

Some basic notions of discrete mathematics

The main purpose of this chapter is to introduce and recall several elementary notions and concepts which will be frequently used in the following. It also helps to fix some important notation. We start off from informal set theory and introduce some basic sets and basic set-theoretic operations. Then we turn to relations and functions and look more carefully at the set of the natural numbers and the principle of complete induction. Afterwards we say a few words about finite, countable, and uncountable sets.

1.1 Basic sets and basic set-theoretic operations

Set theory provides a simple and universal framework for formalizing (in principle) all concepts of mathematics and computer science. Modern set theory goes back to Georg Cantor (1845–1918) and his work on the uncountability of the real line. He also came up with an attempt to characterize sets:

> Unter einer "Menge" verstehen wir jede Zusammenfassung M von bestimmten wohlunterschiedenen Objecten m unsrer Anschauung oder unseres Denkens (welche die "Elemente" von M genannt werden) zu einem Ganzen.

Let us consider this for a moment as a proper definition of set. Then, given a property $\varphi(x)$, we can form the set

$$\{x : \varphi(x)\}$$

consisting of all objects (of our general universe) which satisfy this property. However, in 1901 the English philosopher, logician, mathematician, historian, essayist,

and social critic Bertrand Russell (1872–1970) – Nobel Prize in Literature 1950 – was able to show that this approach is contradictory. To formulate the famous Russell paradox, let

$$R := \{x : x \notin x\},$$

which is a set according to Cantor's characterization. As we can easily see, this implies

$$R \in R \quad \Longleftrightarrow \quad R \notin R.$$

Therefore Cantor's way of characterizing sets leads to a logical contradiction and is thus not acceptable.

After the discovery of Russell's paradox and related paradoxes, a series of formal axiomatic systems for set theory was proposed: Zermelo-Fraenkel set theory ZF, von Neumann-Bernays-Gödel set theory NBG, Morse-Kelley set theory MK just to name a few. The basic underlying idea always is to be more careful – and thus restrictive – in the formation of sets.

Here we do not follow these formal approaches to set theory but stay rather informal – with some care to avoid inconsistencies. In doing this we avail ourselves of the usual semi-formal mathematical language in which the symbols $\neg$ (negation), $\vee$ (or), $\wedge$ (and), $\Longrightarrow$ (implies), $\Longleftrightarrow$ (equivalent), $\exists$ (exists), and $\forall$ (for all) are used as abbreviations.

We assume that there exists something like a *mathematical universe* $\mathbb{U}$ which contains all mathematical objects, in particular all sets, as elements, but is not a set itself. If $\varphi(x)$ is a property, i.e. a formula in our semiformal language, then the expression $\{x : \varphi(x)\}$ denotes the collection of all objects a which satisfy property $\varphi(x)$ (it may be a set, but does not have to be a set in general). The meaning of collection $\{x : \varphi(x)\}$ is provided by the equivalence

$$a \in \{x : \varphi(x)\} \quad \Longleftrightarrow \quad \varphi(a)$$

for any object a. Similarly, if S is a set and $\varphi(x)$ a property, then the collection $\{x \in S : \varphi(x)\}$ comprises those elements a of S which satisfy $\varphi(x)$; i.e.

$$a \in \{x \in S : \varphi(x)\} \quad \Longleftrightarrow \quad a \in S \wedge \varphi(a).$$

Given finitely many objects $b_1, \ldots, b_n$, we often simply write $\{b_1, \ldots, b_n\}$ instead of $\{x : x = b_1 \vee \ldots \vee x = b_n\}$ such that

$$a \in \{b_1, \ldots, b_n\} \quad \Longleftrightarrow \quad a = b_1 \vee \ldots \vee a = b_n.$$

Finally, if we have a formula $\varphi(a, i)$ such that for all natural numbers i,

$$x = b_i \quad \Longleftrightarrow \quad \varphi(x, i),$$

then also the notation $\{b_0, b_1, \ldots\}$ is used. Then

$$a \in \{b_0, b_1, \ldots\} \quad \Longleftrightarrow \quad \varphi(a, i) \text{ for some natural number } i.$$

So, for example, $\{1, 3, 5, \ldots\}$ can be written for the collection of the odd natural numbers and $\{2, 3, 5, 7, \ldots\}$ for the collection of all prime numbers; in the first case the corresponding formula is the formula $(a = 2i + 1)$, in the second case it says that a is the ith prime number.

The most basic principle in set theory is that sets are uniquely determined by their elements and, as a consequence, sets are identical if and only if they have the same elements.

Extensionality principle.

1. For any sets R and S,

$$R = S \quad \Longleftrightarrow \quad \forall x(x \in R \Longleftrightarrow x \in S).$$

2. The collection $\{x : \varphi(x)\}$ is said to be identical to set S if and only if

$$\forall x(x \in S \Longleftrightarrow \varphi(x)).$$

   In this case we call $\{x : \varphi(x)\}$ a set as well.

A set R is a *subset* of set S (or equivalently S is a superset of R) provided that every element of R belongs to S as well,

$$R \subseteq S \quad \Longleftrightarrow \quad \forall x(x \in R \Longrightarrow x \in S).$$

Trivially, this implies that

$$R = S \quad \Longleftrightarrow \quad R \subseteq S \wedge S \subseteq R.$$

Set R is said to be a *proper subset* of S if and only if it is a subset of S but not identical to S,

$$R \subsetneq S \quad \Longleftrightarrow \quad R \subseteq S \wedge R \neq S.$$

Given two sets R and S, the operations of union, intersection, and set-difference are defined as follows:

$$\begin{aligned}
R \cup S &:= \{x : x \in R \vee x \in S\} && \text{(simple union)},\\
\bigcup S &:= \{x : \exists R(R \in S \wedge x \in R)\} && \text{(general union)},\\
R \cap S &:= \{x : x \in R \wedge x \in S\} && \text{(simple intersection)},\\
\bigcap S &:= \{x : \forall R(R \in S \Longrightarrow x \in R)\} && \text{(general intersection)},\\
R \setminus S &:= \{x : x \in R \wedge x \notin S\} && \text{(difference)}.
\end{aligned}$$

Although these operations act on sets, they do not say whether there exist any sets at all. To overcome this problem, all approaches to set theory include principles which claim the existence of a few basic sets and introduce principles to generate new sets from old ones. In the following we briefly list such set existence principles and two further important axioms. For those who are interested in a more detailed discussion of such principles and a broader introduction to (naive) set theory we refer to the textbooks Halmos [6] or Devlin [3].

Set-existence principles

Pairing: For any objects a and b of our mathematical universe, the collection $\{a, b\}$ is a set.

Union: For any set S, the collection $\bigcup S$ is a set.

Separation: If S is a set and $\varphi(x)$ a property, then the collection $\{x \in S : \varphi(x)\}$ is a set.

Natural numbers: The natural numbers (beginning with 0) form a set which is typically denoted by $\mathbb{N}$.

Power set: For any set S, the collection of all subsets of S is a set which is typically denoted by $\wp(S)$, i.e.

$$\wp(S) = \{R : R \subseteq S\}.$$

Collection: If R is a set and $\varphi(x, y)$ a property, and if $(\forall x \in R)\exists y \varphi(x, y)$, then there exists a set S such that

$$(\forall x \in R)(\exists y \in S)\varphi(x, y).$$

Additional axioms

Regularity: If a set S contains an element, then it contains an element which is minimal with respect to the $\in$-relation, i.e.

$$\exists x(x \in S) \implies (\exists y \in S)(\forall z \in S)(z \notin y).$$

Choice: Let S be a set of non-empty sets. Then we can choose a member from each set in S. In other words, there exists a function F defined on S such that, for each $x \in S$, $F(x)$ is an element of x. For the exact definition of *function* see Definition 9 below.

This finishes the build-up of our set theory. As it turns out, all sets needed to develop ordinary mathematics or computer science are available in this (informal) framework. Just to mention a few examples: the set of integers $\mathbb{Z}$, of rationals $\mathbb{Q}$, of reals $\mathbb{R}$, and of complex numbers $\mathbb{C}$ can be defined and proved to exist there.

In order to show how to deal with these notions we prove two simple lemmas; more practice should be gained by working through the exercises. Afterwards we move freely in our set-theoretic universe and assume that all later definitions and constructions are justified on the basis of the set-existence principles introduced above.

Lemma 1

1. *There is exactly one set without any elements; it is denoted by* $\varnothing$.
2. *For any object* a *the collection* $\{a\}$ *is a set.*
3. *If* R *and* S *are sets, then so are the collections* $R \cup S$, $R \cap S$, *and* $R \setminus S$.
4. *If* S *contains at least one set as element, then* $\bigcap S$ *is a set.*

Proof 1. In view of our set existence principles $\mathbb{N}$ is a set, and thus separation tells us that $\{x \in \mathbb{N} : x \neq x\}$ is a set as well. Since every object is identical to itself, we have

$$a \notin \{x \in \mathbb{N} : x \neq x\}$$

for all objects a. Hence we have a set without elements. By the extensionality principle this set has to be unique.

2. For any object a, the collection $\{a, a\}$ is a set by pairing. Since we also have

$$\forall x(x \in \{a, a\} \iff x \in \{a\}),$$

the extensionality principle yields that also $\{a\}$ is a set.

3. Now let R and S be sets. Then by pairing $\{R, S\}$ is a set and by union $\bigcup\{R, S\}$ is a set, as well. Moreover,

$$\forall x(x \in \bigcup\{R, S\} \iff x \in R \cup S),$$

implying that $R \cup S$ is a set by extensionality. Similarly, by definition we have

$$\forall x(x \in R \cap S \iff x \in \{x \in R : x \in S\}).$$

Since $\{x \in R : x \in S\}$ is a set by separation, $R \cap S$ is a set by extensionality. The proof that $R \setminus S$ is a set follows exactly the same line of reasoning.

4. Suppose that a set R is an element of S. Then the definition of $\bigcap S$ implies

$$\forall x(x \in \bigcap S \iff \{x \in R : \forall Q(Q \in S \implies x \in Q)\}).$$

As above, $\{x \in R : \forall Q(Q \in S \implies x \in Q)\}$ is a set by separation and thus $\bigcap S$ by extensionality. □

So given objects a and b, we can form the pair $\{a, b\}$ which by the extensionality principle is identical to $\{b, a\}$. Hence we speak of an unordered pair. Ordered pairs, on the other hand, can be introduced in many different way, one going back to the Polish mathematician and logician Kazimierz Kuratowski (1896–1980).

Definition 2 *Given two objects a and b, the* Kuratowski pair *is defined by*

$$(a, b) := \{\{a\}, \{a, b\}\}.$$

Lemma 3

1. *For any objects a and b the collection (a, b) is a set.*
2. *For all objects a_1, a_2, b_1, b_2 we have*

$$(a_1, a_2) = (b_1, b_2) \iff a_1 = b_1 \wedge a_2 = b_2.$$

Proof 1. Immediate from pairing and the second assertion of the previous lemma.

2. The direction from right to left is obvious. For the converse direction we distinguish two cases:

(i) $a_1 = a_2$. Then we have

$$\{\{a_1\}\} = \{\{a_1\}, \{a_1, a_2\}\} = (a_1, a_2) = (b_1, b_2) = \{\{b_1\}, \{b_1, b_2\}\}.$$

Therefore $\{a_1\} = \{b_1\}$ and $\{a_1\} = \{b_1, b_2\}$, which implies $a_1 = b_1 = b_2$ and by assumption $a_2 = b_2$.

(ii) $a_1 \neq a_2$. Because of our assumption $(a_1, a_2) = (b_1, b_2)$ we know that

$$(*) \qquad \{\{a_1\}, \{a_1, a_2\}\} = \{\{b_1\}, \{b_1, b_2\}\}.$$

Therefore $\{b_1\} = \{a_1\}$ or $\{b_1\} = \{a_1, a_2\}$. However, $\{b_1\} = \{a_1, a_2\}$ implies that $a_1 = b_1 = a_2$, which contradicts our assumption $a_1 \neq a_2$. Hence $\{b_1\} = \{a_1\}$, yielding $a_1 = b_1$.

Also because of (*), $\{b_1, b_2\} = \{a_1\}$ or $\{b_1, b_2\} = \{a_1, a_2\}$. Suppose $\{b_1, b_2\} = \{a_1\}$. Then $a_1 = b_1 = b_2$, and thus

$$\{\{b_1\}, \{b_1, b_2\}\} = \{\{a_1\}, \{a_1, a_1\}\} = \{\{a_1\}, \{a_1\}\} = \{\{a_1\}\}.$$

But then (*) would imply $\{\{a_1\},\{a_1,a_2\}\} = \{\{a_1\}\}$, so that $a_1 = a_2$, contradicting our assumption. Consequently, $\{b_1,b_2\} = \{a_1,a_2\}$.

So far we have $a_1 = b_1$ and $\{a_1,a_2\} = \{b_1,b_2\}$. If a_2 were equal to b_1, then

$$\{a_1,a_2\} = \{a_1,b_1\} = \{a_1,a_1\} = \{a_1\},$$

violating our assumption $a_1 \neq a_2$. This means that $a_2 = b_2$, finishing our proof. □

n-tuples, for any natural number $n \geq 1$ and any objects $a_1,\ldots,a_n$ are defined iteratively by

$$(a_1) := a_1 \qquad \text{and} \qquad (a_1,\ldots,a_n) : ((a_1,\ldots,a_{n-1}),a_n).$$

It is obvious that the n-tuples $(a_1,\ldots,a_n)$ are sets and that

$$(a_1,\ldots,a_n) = (b_1,\ldots,b_n) \iff a_1 = b_1 \wedge \ldots \wedge a_n = b_n.$$

1.2 Relations

With n-tupling at hand we can now easily introduce the important notions of relation and function. To do so, we begin with introducing n-ary Cartesian products and n-ary powers of sets.

Definition 4

1. *Given a natural number $n \geq 1$ and sets $S_1,\ldots,S_n$, the* Cartesian product $S_1 \times \ldots \times S_n$ *is defined by*

$$S_1 \times \ldots \times S_n := \{(x_1,\ldots,x_n) : x_1 \in S_1 \wedge \ldots \wedge x_n \in S_n\}.$$

2. *Given a natural number n and a set S we set*

$$S^n := \begin{cases} \{\varnothing\} & \text{if } n = 0, \\ S \times \ldots \times S \ \ (n\text{-times}) & \text{if } n > 0. \end{cases}$$

Therefore $S_1 \times \ldots \times S_n$ is the collection of all ordered n-tuples whose ith component belongs to S_i $(1 \leq i \leq n)$. It is left to the reader as an exercise to show that $S_1 \times \ldots \times S_n$ is a set.

Given sets $S_1,\ldots,S_n$, the notion of relation is the mathematically precise way to describe specific connections between elements of $S_1,\ldots,S_n$.

Definition 5 *Suppose that n is an arbitrary natural number and $S,S_1,\ldots,S_n$ are arbitrary sets.*

1. *Any $Q \subseteq S^n$ is called an* n-ary relation on S, *and any $Q \subseteq S_1 \times \ldots \times S_n$ is called a* relation on $S_1, \ldots, S_n$.

2. *A set Q is called a* binary relation *if and only if there exist sets R and S such that $Q \subseteq R \times S$.*

3. *For any binary relation Q we define*

$$\mathrm{Dom}(Q) := \{x : \exists y((x,y) \in Q)\} \quad \textit{and} \quad \mathrm{Ran}(Q) := \{y : \exists x((x,y) \in Q)\}.$$

We say that $\mathrm{Dom}(Q)$ *is the* domain *and* $\mathrm{Ran}(Q)$ *the* range *of* Q.

Example 6

1. $\mathrm{Suc} := \{(x, x+1) : x \in \mathbb{N}\}$ is a binary relation on $\mathbb{N}$, the so-called successor relation; $\mathrm{Dom}(\mathrm{Suc}) = \mathbb{N}$ and $\mathrm{Ran}(\mathrm{Suc}) = \{x \in \mathbb{N} : 0 < x\}$.

2. Given any set R, the collection $\{(X,Y) : X \subseteq R \wedge Y \subseteq R \wedge X \subseteq Y\}$ represents the binary subset relation on $\wp(R)$; its domain and its range are the set $\wp(R)$.

3. The ternary relation $\{(x,y,z) \in \mathbb{R}^3 : x^2 + y^2 + z^2 = 1\}$ describes the surface of the unit sphere around the origin $(0,0,0)$.

4. The empty set $\varnothing$ is a relation on any sets $S_1, \ldots, S_n$.

Relations also play an important role in the context of databases, most prominently in connection with the so-called *relational databases*. Relations are then often represented as tables. The columns are the *attributes* and correspond to the factors of the Cartesian product, the elements of the relation are the rows of the table.

Example 7 Let *Names* be the set of names of all students of Bern University, *Faculties* the set of all faculties, and *Courses* the set of all courses offered in HS-2010. Then we can build the table which tells us which student of which faculty takes which courses. This table corresponds to a relation on $\mathit{Names} \times \mathit{Faculties} \times \mathit{Courses}$.

There are three operations on relations which are crucial in database theory: selection, projection, and natural join. In the following we present their mathematical definitions; you should choose your favorite database language, SQL for example, and find out how they are realized there.

Selection Suppose that we are given n attributes $S_1, \ldots, S_n$ and a relation R on $S_1, \ldots, S_n$. In addition, assume $1 \leq i_1 < \ldots i_m \leq n$ and let $\varphi(a_1, \ldots, a_m)$ be an m-ary property. Then we can form the relation

$$Q := \{(x_1, \ldots, x_n) \in R : \varphi(x_{i_1}, \ldots, x_{i_m})\}.$$

Q is called the *selection of R with respect to φ*.

Projection As before, take n attributes $S_1, \ldots, S_n$ and a relation R on $S_1, \ldots, S_n$. For any natural number m with $1 \leq m \leq n$ we can now form the relation

$$\pi_{(1,\ldots,m-1,m+1,\ldots,n)}(R) :=$$
$$\{(x_1, \ldots, x_{m-1}, x_{m+1}, \ldots, x_n) : (\exists y \in S_m)((x_1, \ldots, x_{m-1}, y, x_{m+1}, \ldots, x_n) \in R)\}$$

on $S_1, \ldots, S_{m-1}, S_{m+1}, \ldots, S_n$. It is obtained from (the table of) R by deleting the mth column.

Natural Join Now suppose that Q be a relation on $S_1, \ldots, S_m$ and R a relation on $S_m, \ldots, S_{m+n}$. The *natural join* of Q and R (with respect to the common attribute S_m) is defined as

$$Q \bowtie R :=$$
$$\{(x_1, \ldots, x_m, \ldots, x_{m+n}) : (x_1, \ldots, x_m) \in Q \ \wedge \ (x_m, \ldots, x_{m+n}) \in R)\}.$$

Example 8 Consider four attributes $S_1, \ldots, S_4$ and the relations $Q \subseteq S_1 \times S_2 \times S_3$ and $R \subseteq S_3 \times S_4$ which are given by the following tables:

S_1	S_2	S_3
1	2	3
1	4	5
2	5	3
4	2	6
7	8	5

and

S_3	S_4
3	9
7	2
5	5
3	4

Then $Q \bowtie R$ is represented by the table

S_1	S_2	S_3	S_4
1	2	3	9
1	2	3	4
1	4	5	5
2	5	3	9
2	5	3	4
7	8	5	5

Above we dealt with one-attribute projections and one-attribute natural joins only. However, following the same pattern projections can be defined for several attributes and natural joins with respect to several common attributes.

1.3 Functions

Functions are specific binary relations which assign to any element of their respective domains a unique value. In some cases these values may be determined by carrying through certain computations – as it is the case, for example, if the functions are represented by computer programs. However, the formal concept of a function is only based on this uniqueness condition and does in general not provide a "recipe" how to calculate the values.

Definition 9

1. *A binary relation F is called a* function *if and only if it satisfies the following uniqueness condition:*

$$(\forall x, y, z)((x, y) \in F \wedge (x, z) \in F \implies y = z).$$

2. *A function F is said to be a function from a set R to a set S, in symbols*

$$F : R \to S,$$

if and only if $\mathrm{Dom}(F) = R$ *and* $\mathrm{Ran}(F) \subseteq S$.

3. *If F is a function from set R to set S and $a \in R$, then we often write $F(a)$ for the uniquely determined $b \in S$ such that $(a, b) \in F$; $F(a)$ is the* image (value) *of a under F.*

4. *Given sets R and S, we write S^R for the set of all functions from R to S.*

Often functions F from R to S are introduced by specifying the images under F of all elements x of R by means of expressions $\underline{\ldots x \ldots}$ depending on x which tell us the values $F(x)$:

$$F : R \to S; \quad F(x) := \underline{\ldots x \ldots} \ .$$

There exist many other ways of depicting functions, depending on the respective context, but it should always be clear how they have to be understood.

The empty set $\varnothing$ is a function, and $\mathrm{Dom}(\varnothing) = \mathrm{Ran}(\varnothing) = \varnothing$; it is the only function whose domain is empty. If F is a function from Q to R and S is a superset of R, then F is also a function from Q to S.

For any set R, there always exists the *identity function* id_R on R which maps every element of R to itself,

$$\mathrm{id}_R : R \to R, \quad \mathrm{id}_R(x) := x.$$

Since $\mathrm{id}_R = \{(x, x) : x \in R\}$, we see that identity function of the empty set is the empty set, $\mathrm{id}_\varnothing = \varnothing$.

Example 10

1. Set $F := \{(x, y) : x \in \mathbb{N} \wedge y \in \mathbb{N} \wedge y = x^2\}$. Then F is a function from $\mathbb{N}$ to $\mathbb{N}$; it can also be introduced as

   $$F : \mathbb{N} \to \mathbb{N}; \quad F(x) := x^2.$$

   Of course, F is also a function from $\mathbb{N}$ to any superset of $\mathbb{N}$.

2. $G := \{(x, y) : x \in \mathbb{N} \wedge y \in \mathbb{N} \wedge y = x - 1\}$ is a function from $\mathbb{N} \setminus \{0\}$ to $\mathbb{N}$; however, it is not a function from $\mathbb{N}$ to $\mathbb{N}$ since $0 \notin \mathrm{Dom}(G)$.

3. The relation $H := \{(x, y) : x \in \mathbb{N} \wedge y \in \mathbb{R} \wedge y^2 = x\}$ is not a function since, for example, $(1, 1) \in H$ and $(1, -1) \in H$.

If F is a function from the Cartesian product $R_1 \times \ldots \times R_n$ to a set S, we write $F(a_1, \ldots, a_n)$ instead of $F((a_1, \ldots, a_n))$ and think of F as an n-ary function with arguments $a_1, \ldots, a_n$. Also, very often binary functions are written in infix notation.

Example 11 The function $\dot{-}$ from $\mathbb{N} \times \mathbb{N}$ given by

$$\dot{-} : \mathbb{N} \times \mathbb{N} \to \mathbb{N}; \quad m \dot{-} n := \begin{cases} m - n & \text{if } n \leq m, \\ 0 & \text{if } m < n \end{cases}$$

is the *truncated subtraction*; whenever the "real subtraction" would yield a negative value it gives the value 0.

There are three particularly important classes of functions from a set S to a set T: (i) injective functions (also called one-to-one functions) which never map distinct elements of S to the same element of T, (ii) surjective functions (also called onto functions) which have the property that their range is the set S, and (iii) bijective functions which provide an exact correspondence between S and T. These notions are defined more precisely as follows.

Definition 12 *Let F be a function from R to S.*

1. *F is called an* injective (or one-to-one) function from R to S *if and only if*

$$(\forall x, y \in R)(F(x) = F(y) \implies x = y).$$

2. *F is called a* surjective (or onto) function from R to S *if and only if*

$$(\forall y \in S)(\exists x \in R)(y = F(x)).$$

3. *F is called a* bijective function from R to S *if and only if it is an injective and surjective function from R to S.*

Obviously, the identity function id_R of any set R is a bijective function from R to itself.

Example 13 Let F, G, and H be the functions from $\mathbb{Z}$ to $\mathbb{Z}$ given by

$$F : \mathbb{Z} \to \mathbb{Z}; \quad F(x) := 2x,$$
$$G : \mathbb{Z} \to \mathbb{N}; \quad G(x) := |x|,$$
$$H : \mathbb{Z} \to \mathbb{Z}; \quad H(x) := x + 5.$$

Then F is injective from $\mathbb{Z}$ to $\mathbb{Z}$, but not surjective; G is surjective from $\mathbb{Z}$ to $\mathbb{N}$, but not injective; and H is bijective from $\mathbb{Z}$ to $\mathbb{Z}$.

The following remark mentions a few simple properties in connection with injective and surjective functions. Its proof is left to the reader as an (easy) exercise.

Remark 14 Let R_1, R_2, S_1, S_2 be any sets such that $R_1 \subseteq R_2$ and $S_1 \subseteq S_2$.

1. If there exists an injective function from R_2 to S_1, then there also exists an injective function from R_1 to S_2.

2. If $R_1 \neq \varnothing$ and there exists a surjective function from R_1 to S_2, then there also exists an surjective function from R_2 to S_1.

We now present an important result of set theory. Its formulation is due to Georg Cantor, and it has been proved, independently, by Felix Bernstein and Ernst Schröder. We do not include the proof of this theorem; it can be found in most textbooks on set theory, for example in Deiser [2].

Theorem 15 (Cantor-Schröder-Bernstein)

If F is an injective function from set R to set S and G an injective function from S to R, then there exists a bijective function from R to S.

Function composition is obtained by applying one function to the results of another: if the value $F(a)$ of an element a from the domain of function F belongs to the domain of function G, then there exists the image of $F(a)$ under G, namely $G(F(a))$.

Definition 16 *Suppose we are given a function F from a set R_1 to a set R_2 and a function G a function from a set S_1 to a set S_2; assume, in addition, that $R_2 \subseteq S_1$. Then the* composition $(G \circ F)$ *is the function from R_1 to S_2 given by*

$$(G \circ F) : R_1 \to S_2; \quad (G \circ F)(x) := G(F(x)).$$

Example 17 Let F and G be the functions given by

$$F : \mathbb{N} \to \mathbb{N}; \quad F(x) := 2x,$$
$$G : \mathbb{Z} \to \mathbb{Z}; \quad G(x) := -4x.$$

Then the composition $(G \circ F)$ is the function from $\mathbb{N}$ to $\mathbb{Z}$ such that $(G \circ F)(x) = -8x$ for all natural numbers x.

The identity functions act as neutral elements with respect to function composition: If F is a function form set R to set S, then we have

$$(\mathrm{id}_S \circ F) \;=\; F \;=\; (F \circ \mathrm{id}_R).$$

If F is a bijective function from set R to set S, then it has a unique inverse function F^{-1}, defined by

$$F^{-1} \;:=\; \{(y, x) : y \in S \wedge x \in R \wedge F(x) = y\},$$

which goes from S to R and satisfies $(F^{-1} \circ F) = \mathrm{id}_R$ and $(F \circ F^{-1}) = \mathrm{id}_S$.

Let F be a function from a set Q_1 to a set Q_2, G a function from a set R_1 to a set R_2, and H a function from a set S_1 to a set S_2. Assume, in addition, that $Q_2 \subseteq R_1$ and $R_2 \subseteq S_1$. Then we can form the compositions $(H \circ (G \circ F))$ and $((H \circ G) \circ F)$, both being functions from Q_1 to S_2, and, as is easily checked,

$$(H \circ (G \circ F)) \;=\; ((H \circ G) \circ F)).$$

Therefore we often simply write $(H \circ G \circ F)$ for this function.

For any natural number $n \geq 1$ and any function from set S to set S the nth power of F is the function $(F)^n$ from S to S iteratively defined as follows:

$$(F)^1 := F \qquad \text{and} \qquad (F)^n := (F \circ (F)^{n-1}).$$

For example, $(F)^3(a) = F(F(F(a)))$.

Remark 18

1. If F is a surjective function from Q to R and G a surjective function from R to a set S, then $(G \circ F)$ is a surjective function from Q to S.

2. If F is an injective function from Q to R_1 and G an injective function from R_2 to a set S and if $R_1 \subseteq R_2$, then $(G \circ F)$ is an injective function from Q to S.

The proof of these properties is straightforward and only requires applications of the definitions of surjectivity and injectivity.

1.4 The set $\mathbb{N}$, recursion, and complete induction

In this section we look more closely at the set of the natural numbers, with particular emphasis on recursive definitions and the principle of complete induction. Recursive definitions are one of the most important definition principles in mathematics and computer science, complete induction is the corresponding and equally important proof principle.

Informally written, $\mathbb{N}$ is the set $\{0, 1, 2, \ldots\}$, and its elements are called *natural numbers*. $\mathbb{N}$ can be characterized as the least collection of objects which contains the number 0 and is closed under successor, i.e.

$$\mathbb{N} = \bigcap\{X : 0 \in X \wedge \forall x(x \in X \Longrightarrow x + 1 \in X)\}.$$

This characterization implies that any set S of natural numbers which has the following two properties

$$0 \in S \qquad \text{and} \qquad (\forall x \in S)(x + 1 \in S) \ \text{(saying that } S \text{ is closed under successor)}$$

contains all natural numbers. This principle of generating the natural numbers from 0 by the successor operation is also the core of recursive definitions.

Suppose that we are given an arbitrary non-empty set S. We are interested in functions F from $\mathbb{N}$ to S and observe that there are at least two possibilities to define such functions:

(P1) We have an expression $t(n)$ which, for any natural number n, tells us the value of $F(n)$. To give an example, take $F(n)$ to be the nth digit in the decimal notation of π; $F(0) = 3$, $F(1) = 1$, $F(2) = 4$, $F(3) = 1$, $F(4) = 5$, ...

(P2) The first value $F(0)$ is given explicitly, and there exists a rule which tells us how to determine the value of $F(n+1)$ based on the value of $F(n)$. An example of that approach is the usual definition of the factorial:

$$\begin{aligned} 0! &= 1, \\ (n+1)! &= (n+1) \cdot n! \end{aligned}$$

The form of (P2) is the prototype of the most simple way in which functions from the natural numbers to a given set are defined by recursion.

Definition 19 *Let S be a non-empty set, $m \in S$, and $G : S \times \mathbb{N} \to S$. Then the function $F : \mathbb{N} \to S$ is* recursively *defined from (m, G) in case that*

$$\begin{aligned} F(0) &= m, \\ F(n+1) &= G(F(n), n). \end{aligned}$$

This definition principle can be generalized in many ways. Rather than going into the formal details, we present a few typical examples.

Example 20 (Fibonacci)

In the recursive definition of the *Fibonacci function* $\mathrm{Fib} : \mathbb{N} \to \mathbb{N}$ we explicitly fix the first two values and let the latter depend on its two immediate predecessors:

$$\begin{aligned} \mathrm{Fib}(0) &= 0, \\ \mathrm{Fib}(1) &= 1, \\ \mathrm{Fib}(n+2) &= \mathrm{Fib}(n) + \mathrm{Fib}(n+1). \end{aligned}$$

Example 21 (Ackermann)

The standard definition of the *Ackermann* function $\mathrm{Ack} : \mathbb{N}^2 \to \mathbb{N}$ is a typical example of *nested recursion*. We have nested calls to previously defined values:

$$\begin{aligned} \mathrm{Ack}(m, 0) &= 0, \\ \mathrm{Ack}(0, n+1) &= 2(n+1), \\ \mathrm{Ack}(m+1, 1) &= 2, \\ \mathrm{Ack}(m+1, n+2) &= \mathrm{Ack}(m, \mathrm{Ack}(m+1, n+1)). \end{aligned}$$

The Ackermann function is a rapidly growing function; actually, it is a recursive (computable) function which grows faster than any primitive recursive function.

Example 22 (Fast growing hierarchy)
Consider the family of functions $(\mathrm{FH}_n)_{n\in\mathbb{N}}$, all from $\mathbb{N}$ to $\mathbb{N}$, defined by, for any $m \in \mathbb{N}$,

$$\begin{aligned} \mathrm{FH}_0(m) &= m+1, \\ \mathrm{FH}_{n+1}(m) &= (\mathrm{FH}_n)^m(m). \end{aligned}$$

Here recursion is applied in the form that for determining function FH_{n+1}, we refer to the previously defined function FH_n. If we consider the diagonalization

$$\mathrm{FH}_* : \mathbb{N} \to \mathbb{N}; \quad \mathrm{FH}_*(m) := \mathrm{FH}_m(m)$$

we have another example of a (very) fast growing function; as the Ackermann function, FH_* is recursive but not primitive recursive.

The characterization of the set of natural numbers as the least set which contains 0 and is closed under successor is directly related to the principle of complete induction (sometimes also denoted as the principle of mathematical induction).

Complete induction. For all formulas $\varphi(a)$ we have

$$\varphi(0) \wedge (\forall n \in \mathbb{N})(\varphi(n) \Longrightarrow \varphi(n+1)) \implies (\forall n \in \mathbb{N})\varphi(n).$$

Complete induction provides a proof principle. In order to show that a property $\varphi(a)$ holds for all natural numbers, it is sufficient to show

(i) induction base: $\varphi(0)$,

(ii) induction step: $\varphi(n) \Longrightarrow \varphi(n+1)$ for any natural number n.

The premise $\varphi(n)$ in (ii) is often referred to as the induction hypothesis (IH). In the following we will often make use of this form of complete induction and variants which we will introduce later. We begin with a simple number-theoretic property.

Lemma 23 *For all natural numbers n we have $n < 2^n$.*

Proof We show this assertion by complete induction.

Induction base: $0 < 1 = 2^0$.

Induction step: Given a natural number n, the IH states $n < 2^n$, and from that we obtain

$$n+1 < 2^n + 1 = 2^n + 2^0 \leq 2^n + 2^n = 2^n \cdot 2^1 = 2^{n+1}.$$

Therefore, if $\varphi(n)$ is the formula $n < 2^n$, then we have shown

$$\varphi(0) \wedge (\forall n \in \mathbb{N})(\varphi(n) \Longrightarrow \varphi(n+1)).$$

By complete induction we thus obtain $(\forall n \in \mathbb{N})\varphi(n)$, as desired. □

A first variant of complete induction gives us the freedom to work on a surjective image of the natural numbers. In a second variant the induction hypothesis is available for all predecessors of a given number; for its formulation it is convenient to define, for all $m \in \mathbb{N}$,

$$\mathbb{N}_m := \{n \in \mathbb{N} : n < m\}.$$

This form of complete induction is sometimes called $<$-*induction*.

Lemma 24 (Variants of complete induction)

1. *Let M be a set and F a surjective function from $\mathbb{N}$ to M. For all formulas $\varphi(a)$ we then have*

$$\varphi(F(0)) \wedge (\forall n \in \mathbb{N})(\varphi(F(n)) \Longrightarrow \varphi(F(n+1))) \implies (\forall x \in M)\varphi(x).$$

2. *$<$-induction: For all formulas $\varphi(a)$ we have*

$$(\forall m \in \mathbb{N})((\forall n \in \mathbb{N}_m)\varphi(n) \Longrightarrow \varphi(m)) \implies (\forall m \in \mathbb{N})\varphi(m).$$

Proof 1. We assume

$$\varphi(F(0)), \tag{1}$$

$$(\forall n \in \mathbb{N})(\varphi(F(n)) \Longrightarrow \varphi(F(n+1))) \tag{2}$$

and set $\chi(n) := \varphi(F(n))$. Then we have

$$\chi(0) \tag{3}$$

because of (1) and

$$(\forall n \in \mathbb{N})(\chi(n) \Longrightarrow \chi(n+1)) \tag{4}$$

because of (2). In view of the principle of complete induction we can deduce from (3) and (4) that $(\forall n \in \mathbb{N})\chi(n)$. However, by the surjectivity of F and the definition of $\chi(n)$ this implies $(\forall x \in M)\varphi(x)$.

2. Now we have the assumption

$$(\forall m \in \mathbb{N})((\forall n \in \mathbb{N}_m)\varphi(n) \Longrightarrow \varphi(m)) \tag{5}$$

and set $\psi(m) \coloneqq (\forall n \in \mathbb{N}_m)\varphi(n)$. Obviously,

$$\psi(0) \tag{6}$$

since there are no natural numbers smaller than 0. Also, if $\psi(m)$, then (5) implies $\varphi(m)$. Observe that therefore $\psi(m)$ yields $(\psi(m) \wedge \varphi(m))$, i.e.

$$\psi(m) \implies \psi(m+1). \tag{7}$$

Because of (6) and (7), complete induction gives us $(\forall m \in \mathbb{N})\psi(m)$. The desired assertion $(\forall m \in \mathbb{N})\varphi(m)$ is an immediate consequence. □

Example 25

1. Let m be a natural number and set $\mathbb{N}_{\geq m} \coloneqq \{n \in \mathbb{N} : m \leq n\}$. For all formulas $\varphi(a)$ we then have

   $$\varphi(m) \wedge (\forall n \in \mathbb{N}_{\geq m})(\varphi(n) \implies \varphi(n+1)) \implies (\forall n \in \mathbb{N}_{\geq m})\varphi(n).$$

   This follows immediately from the previous lemma by setting

   $$F : \mathbb{N} \to \mathbb{N}_{\geq m}; \quad F(n) \coloneqq m + n.$$

2. Let m be a natural number and set $m\mathbb{N} \coloneqq \{(m \cdot n) : n \in \mathbb{N}\}$. For all formulas $\varphi(a)$ we then have

   $$\varphi(0) \wedge (\forall n \in m\mathbb{N})(\varphi(n) \implies \varphi(n+m)) \implies (\forall n \in m\mathbb{N})\varphi(n).$$

   This follows immediately from the previous lemma by setting

   $$F : \mathbb{N} \to m\mathbb{N}; \quad F(n) \coloneqq m \cdot n.$$

3. For all natural numbers n we have

   $$\left(\frac{3}{2}\right)^{n+2} \leq \mathrm{Fib}(n+4).$$

   To establish this assertion by <-induction we show, for all natural numbers n, that

   $$(\forall m \in \mathbb{N}_n)\left(\left(\frac{3}{2}\right)^{m+2} \leq \mathrm{Fib}(m+4)\right) \implies \left(\frac{3}{2}\right)^{n+2} \leq \mathrm{Fib}(n+4). \tag{1}$$

   So we fix a natural number n, assume

   $$(\forall m \in \mathbb{N}_n)\left(\left(\frac{3}{2}\right)^{m+2} \leq \mathrm{Fib}(m+4)\right), \tag{2}$$

and distinguish the following cases:

(i) $n = 0$. Then trivially

$$\left(\frac{3}{2}\right)^{n+2} = \left(\frac{3}{2}\right)^{2} = \frac{9}{4} < 3 = \mathrm{Fib}(4) = \mathrm{Fib}(n+4).$$

(ii) $n = 1$. Then also trivially

$$\left(\frac{3}{2}\right)^{n+2} = \left(\frac{3}{2}\right)^{3} = \frac{27}{8} < 5 = \mathrm{Fib}(5) = \mathrm{Fib}(n+4).$$

(iii) $n \geq 2$. In this case we have in view of (2) that

$$\begin{aligned}\left(\tfrac{3}{2}\right)^{n+2} &= \left(\tfrac{3}{2}\right)^{n}\left(\tfrac{3}{2}\right)^{2} < \left(\tfrac{3}{2}\right)^{n}\left(1+\tfrac{3}{2}\right) = \left(\tfrac{3}{2}\right)^{n} + \left(\tfrac{3}{2}\right)^{n+1} \\ &\leq \mathrm{Fib}(n+2) + \mathrm{Fib}(n+3) = \mathrm{Fib}(n+4).\end{aligned}$$

Therefore $\left(\frac{3}{2}\right)^{n+2} \leq \mathrm{Fib}(n+4)$ has been shown for all natural numbers n, completing the proof of (1).

Theorem 26 (Least element)
Every non-empty set of natural numbers has a least element; i.e. for all $S \subseteq \mathbb{N}$ we have

$$S \neq \varnothing \quad \Longrightarrow \quad (\exists m \in S)(\forall n \in S)(m \leq n).$$

This least element of S is often denoted by $\min(S)$.

Proof In view of Lemma 24.2 we know that

$$(\forall m \in \mathbb{N})((\forall n \in \mathbb{N}_m)(n \notin S) \Longrightarrow m \notin S) \quad \Longrightarrow \quad (\forall m \in \mathbb{N})(m \notin S).$$

By a simple logical transformation this yields

$$(\exists m \in \mathbb{N})(m \in S) \quad \Longrightarrow \quad (\exists m \in \mathbb{N})((\forall n \in \mathbb{N}_m)(n \notin S) \wedge m \in S).$$

Now, since S is supposed to be a subset of $\mathbb{N}$, this assertion can be simplified to

$$S \neq \varnothing \quad \Longrightarrow \quad (\exists m \in S)(\forall n \in \mathbb{N}_m)(n \notin S),$$

which is exactly what we wanted to prove. □

1.5 Finite, countably infinite, and uncountable sets

We want now to classify sets according to their size. Actually, we confine ourselves to distinguish between finite, countably infinite, and uncountable sets. In a more advanced setting also the uncountable sets can be graded, but this is beyond the scope of these notes.

Definition 27 *Let S be an arbitrary set.*

1. *S is called* finite *if there exists a natural number n and a surjective function from $\mathbb{N}_n$ to S. If S is not finite it is said to be* infinite*.*

2. *S is called* countably infinite *if it is infinite and there exists a surjective function from $\mathbb{N}$ to S.*

3. *S is called* countable *if it is finite or countably infinite. If S is not countable, it is said to be uncountable.*

Hence a set S is countable if and only if there exists a surjective function from $\mathbb{N}$ to S and uncountable if and only if such a surjection does not exist. The empty set $\varnothing$ is finite since $\varnothing$ is a surjective function from $\mathbb{N}_0$ to $\varnothing$.

Lemma 28

1. *If R is a finite set and if there exists a surjective function from R to a set S, then S is finite as well.*

2. *If S is a finite set and if there exists an injective function from a set R to S, then R is finite as well.*

3. *If there exists a bijective function from a set R to a set S, then both R and S are finite or both R and S are infinite.*

PROOF 1. By assumption, there exist a natural number n, a surjective function F from $\mathbb{N}_n$ to R, and a surjective function G from R to S. Therefore, $(G \circ F)$ is a surjective function from $\mathbb{N}_n$ to S. Thus S is finite.

2. The proof of the second assertion is similar and left to the reader as an exercise.

3. The third assertion is immediate from the first and the second. □

Lemma 29 *If m and n are natural numbers and F is a surjective function from $\mathbb{N}_m$ to $\mathbb{N}_n$, then $n \leq m$.*

PROOF Assume that our assertion is not correct. Then by Theorem 26 there exists a least natural number m for which we have a natural number $n > m$ and a surjective function F from $\mathbb{N}_m$ to $\mathbb{N}_n$.

We first observe that $m > 0$. Otherwise, $\mathbb{N}_m = \varnothing$, implying, by the surjectivity of F, that $\mathbb{N}_n = \varnothing$ as well and thus $n = 0$, contradicting our assumption $n > m$.

So $m = m_0 + 1$ and $n = n_0 + 1$ for natural numbers $n_0 > m_0$. Now consider the following two cases:

(i) $F(m_0) = n_0$. Then define

$$G : \mathbb{N}_{m_0} \to \mathbb{N}_{n_0}; \quad G(k) := \begin{cases} F(k) & \text{if } F(k) < F(m_0), \\ 0 & \text{if } F(k) = F(m_0). \end{cases}$$

(ii) $F(m_0) < n_0$. In this case define

$$G : \mathbb{N}_{m_0} \to \mathbb{N}_{n_0}; \quad G(k) := \begin{cases} F(k) & \text{if } F(k) \leq F(m_0), \\ F(k) - 1 & \text{if } F(k) > F(m_0). \end{cases}$$

It is not hard to verify that G is a surjective function from $\mathbb{N}_{m_0}$ to $\mathbb{N}_{n_0}$ in both cases.

However, this contradicts our choice of m, and so it is not possible that there exists a surjective function from the predecessors of a natural number to the predecessors of a larger natural number. □

Corollary 30 $\mathbb{N}$ *is a countably infinite set.*

PROOF Assume that there exists a surjective function from $\mathbb{N}_n$ for some natural number n to $\mathbb{N}$. But then $\mathbb{N}_n$ cannot be the empty set, and by Remark 14.2 there exists a surjective function from $\mathbb{N}_n$ to $\mathbb{N}_{n+1}$, contradicting the previous lemma. Hence $\mathbb{N}$ is infinite, and so it is clear that $\mathbb{N}$ is countably infinite. □

If S is a finite set then by this definition there exists a natural number n and a surjective function from $\mathbb{N}_n$ to S. This means that the set

$$\{n \in \mathbb{N} : \exists F(F \text{ surjective function from } \{m \in \mathbb{N} : m < n\} \text{ to } S)\}$$

is a non-empty subset of the natural numbers and has a least element according to Theorem 26. Therefore the following definition makes sense.

Definition 31 *For any finite set S the* size $|S|$ *of S is defined by*

$$|S| := \min(\{n \in \mathbb{N} : \exists F(F \textit{ surjective function from } \mathbb{N}_n \textit{ to } S)\}).$$

If S is an infinite set, then we set $|S| := \infty$; and we stipulate that $n < \infty$ for all natural numbers n.

So, if S is a finite set, then the size of S simply is the number of its elements; the size of any infinite set is defined to be ∞. In proper set theory one does not speak of the size of set but of its cardinality. On finite sets, size and cardinality agree; on infinite sets, on the other hands, cardinalities allow us to distinguish between different degrees of infinity. For more details see, for example, [2, 3, 6].

Lemma 32 *If S is a finite set and $n = |S|$, then there exists a bijective function from $\mathbb{N}_n$ to S.*

PROOF We know from the previous definition that there exists a surjective function F from $\mathbb{N}_n$ to S. If F is not bijective, then we have $m_1 < m_2 < n$ and $F(m_1) = F(m_2)$ for some natural numbers m_1 and m_2; also $2 \leq n$. Now set

$$G : \mathbb{N}_{n-1} \to S; \quad G(m) := \begin{cases} F(m) & \text{if } m < m_2, \\ F(m+1) & \text{if } m_2 \leq m. \end{cases}$$

Then G is a surjective function from $\mathbb{N}_{n-1}$ to S, contradicting the minimality of $n = |S|$. Therefore F has to be bijective. □

From what we have seen so far the following assertions about the sizes of some specific sets are obvious.

Remark 33

1. $|\varnothing| = 0$ and $|\mathbb{N}| = \infty$.
2. For all objects a we have $|\{a\}| = 1$.
3. For all natural numbers n we have $|\mathbb{N}_n| = n$.
4. If R is a subset of S, then $|R| \leq |S|$.

Theorem 34 *If there exists a bijective function from set R to set S, then $|R| = |S|$.*

PROOF From Lemma 28.3 we know already that R and S are both finite or both infinite. Hence for the infinite case we have $|R| = \infty = |S|$.

Now suppose that R and S are finite. By Lemma 32 there exist a bijective function F from $\mathbb{N}_{|R|}$ to R and a bijective function G from $\mathbb{N}_{|S|}$ to S. By assumption, we also have a bijective function H from R to S. Hence $(G^{-1} \circ H \circ F)$ is a surjective function from $\mathbb{N}_{|R|}$ to $\mathbb{N}_{|S|}$ and $(F^{-1} \circ H^{-1} \circ G)$ a surjective function from $\mathbb{N}_{|S|}$ to $\mathbb{N}_{|R|}$. Lemma 29 implies $|R| = |S|$. □

1.6 Elementary combinatorics

Many simple combinatorial properties have to do with finite sets; some of those will be considered first. Then we turn to countably infinite sets and examples of uncountable sets.

Lemma 35 *For all finite sets R and S:*

1. $|R \cup S| \leq |R| + |S|$.
2. $R \cap S = \varnothing \implies |R \cup S| = |R| + |S|$.

Proof 1. We set $m := |R|$ and $n := |S|$ and pick a bijective function F from $\mathbb{N}_m$ to R as well as a bijective function G from $\mathbb{N}_n$ to S. Then set

$$H : \mathbb{N}_{m+n} \to R \cup S; \quad H(k) := \begin{cases} F(k) & \text{if } k < m, \\ G(k-m) & \text{if } k \geq m. \end{cases}$$

Obviously, H is a surjective function from $\mathbb{N}_{m+n}$ to $R \cup S$. Hence $|R \cup S| \leq m + n$.

2. If $R \cap S = \varnothing$, then H is even bijective. Then by Lemma 29 there cannot exist a natural number $j < m + n$ and a surjective function form $\mathbb{N}_j$ to $R \cup S$. This verifies that $|R \cup S| = |R| + |S|$ in this case. □

Theorem 36 *For all finite sets R and S:*

1. $|R \times S| = |R| \cdot |S|$.
2. $|S^R| = |S|^{|R|}$.
3. $|\wp(S)| = 2^{|S|}$.

Proof 1. We prove the first assertion by complete induction on $|R|$.

$|R| = 0$. Then $R = \varnothing$ and $R \times S = \varnothing$. It follows directly that

$$|R \times S| = 0 = 0 \cdot |S| = |R| \cdot |S|.$$

$|R| > 0$. Then there exists an element $a \in R$, and for $Q := R \setminus \{a\}$ we have $|Q| + 1 = |R|$. The IH implies $|Q \times S| = |Q| \cdot |S|$. Also, it is easy to see that $|\{a\} \times S| = |S|$. Since

$$R \times S = (Q \times S) \cup (\{a\} \times S) \qquad \text{and} \qquad (Q \times S) \cap (\{a\} \times S) = \varnothing,$$

the second part of the previous lemma gives us

$$|R \times S| = (|Q| \cdot |S|) + |S| = (|Q| + 1) \cdot |S| = |R| \cdot |S|.$$

2. The second assertion is proved by induction on R, too.

$|R| = 0$. Then we have

$$|S^R| \;=\; |S^{\varnothing}| \;=\; 1 \;=\; |S|^0 \;=\; |S|^{|R|}.$$

$|R| > 0$. As before, we pick an element $a \in R$ and set $Q := R \setminus \{a\}$ such that $|Q| + 1 = |R|$. Hence the IH implies $|S^Q| = |S|^{|Q|}$. Furthermore, it is not difficult to verify that the function

$$F \;:\; S^Q \times S \;\to\; S^R; \quad F(G,b) \;:=\; G \cup \{(a,b)\}$$

is bijective from $S^Q \times S$ to S^R. From Theorem 34 and the first part of the present theorem we conclude

$$|S^R| \;=\; |S^Q \times S| \;=\; |S^Q| \cdot |S|.$$

Altogether,

$$|S^R| \;=\; |S^Q| \cdot |S| \;=\; |S|^{|Q|} \cdot |S| \;=\; |S|^{|Q|+1} \;=\; |S|^{|R|}.$$

3. The third assertion could be proved by complete induction as well. Alternatively, introduce for any $Q \subseteq S$ the characteristic function Char_Q, defined by

$$\mathrm{Char}_Q \;:\; S \;\to\; \{0,1\}; \quad \mathrm{Char}_Q(x) \;:=\; \begin{cases} 1 & \text{if } x \in Q, \\ 0 & \text{if } x \notin Q. \end{cases}$$

Then the function

$$F \;:\; \wp(S) \;\to\; \{0,1\}^S; \quad F(Q) \;=\; \mathrm{Char}_Q$$

is easily checked to be bijective from $\wp(S)$ to $\{0,1\}^S$. Making use of Theorem 34 and the second part of our present theorem, we see that $|\wp(S)| = |\{0,1\}^S| = 2^{|S|}$, as desired. □

Lemma 37 *If R is a finite set and S a proper subset of R, then $|S| < |R|$.*

Proof Since S is a proper subset of R, there exists an object $a \in R \setminus S$. it follows from Lemma 35.2 and Remark 33.4 that $|S| < |S| + 1 = |S \cup \{a\}| \leq |R|$. □

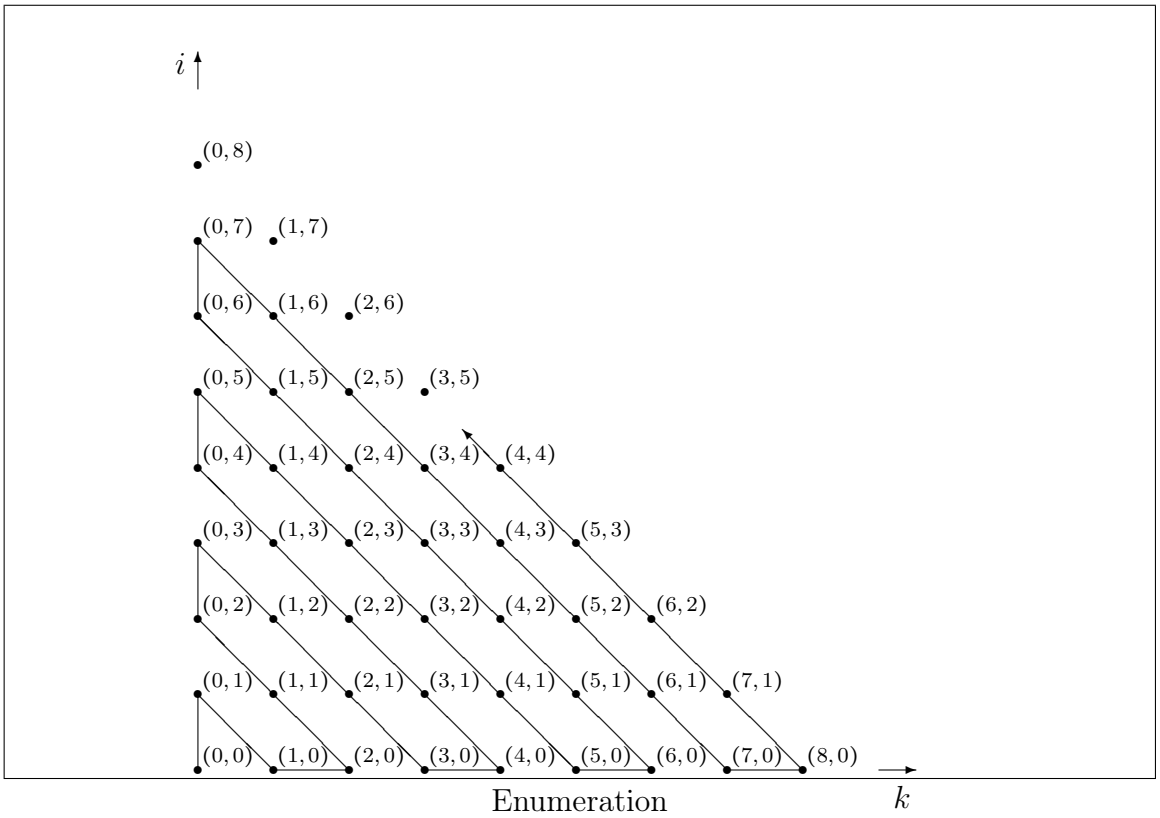
This property is trivial for finite sets and false for infinite sets; consider, for example, the sets $\mathbb{N}$ and $\mathbb{Z}$. Indeed, it is characteristic of finite sets that they cannot be bijectively mapped on proper subsets.

Now we leave the finite sets and say a few words about infinite sets, beginning with the following important observation.

Theorem 38

1. *There exists a bijective function from* $\mathbb{N}$ *to* $\mathbb{N}^2$*. This implies, in particular, that* $\mathbb{N}^2$ *is countably infinite.*
2. *If the sets* R *and* S *are countable, then so is* $R \times S$*.*
3. *If a set* S *is countable, then so is* S^n *for any* $n \in \mathbb{N}$*.*

PROOF 1. The figure below provides an explicit bijective function from $\mathbb{N}$ to $\mathbb{N}^2$; one simply has to follow the line.



2. By assumption we have a surjective function F from $\mathbb{N}$ to R and a surjective function G from $\mathbb{N}$ to S. Define

$$H : \mathbb{N}^2 \to R \times S; \quad H(m,n) := (F(m), G(n)).$$

It is trivial to check that H is a surjective function from $\mathbb{N}^2$ to $R \times S$. The first part of this theorem tells us that there exists a surjective function I from $\mathbb{N}$ to $\mathbb{N}^2$, and by Remark 18.1 the composition $(H \circ I)$ of H and I is a surjective function from $\mathbb{N}$ to $R \times S$, establishing the countability of $R \times S$.

3. The third part of this theorem is proved by a straightforward complete induction on n, making use of the second part in the induction step. □

Every function F with domain $\mathbb{N}$ may be regarded as the countably infinite sequence

$$(F(0), F(1), F(2), \ldots) \;=\; (F(n))_{n\in\mathbb{N}}.$$

Or, to put it the other way round, any countably infinite sequence of sets

$$(S_0, S_1, S_2, \ldots) \;=\; (S_n)_{n\in\mathbb{N}}$$

can be identified with the function F whose domain is the set $\mathbb{N}$ and whose values are given by $F(n) = S_n$ for all natural numbers n.

Given such a countably infinite sequence $(S_n)_{n\in\mathbb{N}}$, we can form the union of its components,

$$\bigcup_{n\in\mathbb{N}} S_n \;:=\; \bigcup\{S_n : n \in \mathbb{N}\}$$

which, clearly, implies, for any object a,

$$a \in \bigcup_{n\in\mathbb{N}} S_n \quad\Longleftrightarrow\quad (\exists n \in \mathbb{N})(a \in S_n).$$

The following theorem formulates a further important closure property of the countably infinite sets.

Theorem 39 *The countable union of countable sets is countable. More precisely: If $(S_n)_{n\in\mathbb{N}}$ is a sequence of countable sets S_n, then the union $\bigcup_{n\in\mathbb{N}} S_n$ is also countable.*

Proof By assumption, every set S_n, for $n \in \mathbb{N}$, is countable, which means that there exists a surjective function F_n from $\mathbb{N}$ to S_n. We define a function G,

$$G \;:\; \mathbb{N}\times\mathbb{N} \;\to\; \bigcup_{n\in\mathbb{N}} S_n; \quad G(m,n) \;:=\; F_n(m).$$

We claim that G is surjective: Given an $a \in \bigcup_{n\in\mathbb{N}} S_n$, there must be an $n_0 \in \mathbb{N}$ such that $a \in S_{n_0}$. Furthermore, since F_{n_0} is surjective, there is an $m_0 \in \mathbb{N}$ such that $a = F_{n_0}(m_0)$. Hence $a = G(m_0, n_0)$.

As in the proof of the second part of the previous theorem, this implies that $\bigcup_{n\in\mathbb{N}} S_n$ is countable. □

Corollary 40 *The sets $\mathbb{Z}$ and $\mathbb{Q}$ as well as all sets $\mathbb{N}^n$, $\mathbb{Z}^n$, and $\mathbb{Q}^n$ with $n \in \mathbb{N}$ are countably infinite.*

Given any set S, we write S^* for the set of all finite sequences with components from S; i.e.

$$S^* := \bigcup_{n\in\mathbb{N}} S^n.$$

Corollary 41 *If S is a countable set, then so is S^*.*

PROOF Simply apply Theorem 38.3 and Theorem 39. □

To end this section, we show that not all sets are countable. In the proof of this statement we make use of *Cantor's diagonalization method*, an important technique which is also applied in other contexts.

Theorem 42 *The sets $\{0,1\}^{\mathbb{N}}$, $\mathbb{N}^{\mathbb{N}}$, $\wp(\mathbb{N})$, and $\mathbb{R}$ are uncountable.*

PROOF We confine ourselves to showing the uncountability of $\{0,1\}^{\mathbb{N}}$; the other sets can be treated similarly.

We set $S := \{0,1\}^{\mathbb{N}}$ and assume that S is countable. Then there exists a surjective function F from $\mathbb{N}$ to S. Accordingly, for any $n \in \mathbb{N}$, $F(n)$ is a function from $\mathbb{N}$ to $\{0,1\}$. Therefore, it is justified to define a further function G,

$$G : \mathbb{N} \to \{0,1\}; \quad G(n) := 1 - F(n)(n).$$

G is an element of S, and by the surjectivity of F there must exist an $n_0 \in \mathbb{N}$ such that $G = F(n_0)$. But then

$$F(n_0)(n_0) = G(n_0) = 1 - F(n_0)(n_0).$$

This is a contradiction, and so the uncountability of S is established. □

The previous results have an immediate and very important consequence for computer science: Choose your favorite programming language PRLA. The alphabet Al(PRLA) of PRLA will be a finite or at most countably infinite set of symbols, and any PRLA-program, therefore, can be regarded as a finite sequence of elements of Al(PRLA) and thus belongs to Al(PRLA)*. Since Al(PRLA)* is countable according to Corollary 41, there exist only countably many PRLA-programs. On the other hand, Theorem 42 tells us that there are uncountably many functions from $\mathbb{N}$ to $\mathbb{N}$. Hence for any programming language there will always be a function from the natural numbers to the natural number which cannot be computed by this language.

1.7 The Euclidian algorithm

The Euclidian algorithm, which goes back to Euclid and was first presented around 300 BC, is an efficient method to compute the greatest common divisor of two

natural numbers greater than 0. It is an important algorithm in elementary number theory which has interesting consequences, for example Bezout's identity. Also, it is used in connection with the famous RSA-algorithm in cryptography, which we will discuss in Section 1.9.

Definition 43 *Let a and b be any integers.*

1. *b is called a* divisor *of a if there exists an integer x such that $a = xb$;*

$$b \mid a \quad :\Longleftrightarrow \quad (\exists x \in \mathbb{Z})(a = xb).$$

   In this case we also say that a is divisible *by b.*

2. *$\mathrm{Div}(a, b)$ is defined to be the set of all common divisors of a and b;*

$$\mathrm{Div}(a, b) \;:=\; \{x \in \mathbb{Z} : x \mid a \text{ and } x \mid b\}.$$

3. *The* greatest common divisor *of a and b is defined as*

$$\gcd(a, b) \;:=\; \begin{cases} \max \mathrm{Div}(a, b) & \text{if } a \neq 0 \text{ or } b \neq 0, \\ 0 & \text{if } a = b = 0. \end{cases}$$

If a is a positive integer and b a divisor of a, then $b \leq a$. For all $a, b, c, x, y \in \mathbb{Z}$ we clearly have

$$c \mid a \wedge c \mid b \quad \Longrightarrow \quad c \mid (xa + yb).$$

Also, $\gcd(a, 0) = a$ for all positive integers a and $\gcd(p, q) = 1$ if p and q are two different prime numbers.

Lemma 44 *For all integers a, b, c, x we have*

$$a = xb + c \quad \Longrightarrow \quad \mathrm{Div}(a, b) = \mathrm{Div}(b, c) \;\wedge\; \gcd(a, b) = \gcd(b, c).$$

Proof Assume $a = xb + c$. If y is an element of $\mathrm{Div}(b, c)$, then it divides b and c. By what we remarked above it therefore also divides a. Hence $\mathrm{Div}(b, c) \subseteq \mathrm{Div}(a, b)$.

On the other hand, our assumption can be rewitten as $c = a - xb$. If y is an element of $\mathrm{Div}(a, b)$, then it divides a and b. Again by what we remarked above it therefore also divides c. Hence $\mathrm{Div}(a, b) \subseteq \mathrm{Div}(b, c)$.

So we have shown that $\mathrm{Div}(a, b) = \mathrm{Div}(b, c)$ follows from $a = xb + c$. This immediately implies $\gcd(a, b) = \gcd(b, c)$. □

Euclidian algorithm EA

INPUT: Positive natural numbers a and b.

Now compute natural numbers $k, q_0, \ldots, q_{k-1}, r_0, \ldots, r_{k+1}$ such that

(EA.1) $r_0 = a \quad \text{and} \quad r_1 = b,$

(EA.2) $r_1 > r_2 > \ldots > r_{k+1} = 0,$

(EA.3) $r_n = q_n r_{n+1} + r_{n+2} \qquad (\text{for } n = 0, \ldots, k-1).$

OUTPUT: r_k.

Theorem 45 *If we input the two positive natural numbers a and b into the Euclidian algorithm* EA*, then* EA *terminates and its output is the greatest common divisor* $\gcd(a,b)$ *of a and b.*

PROOF Termination is obvious since there is no infinite descending sequence of natural numbers. Furthermore, by the previous lemma we have that

$$\gcd(a,b) = \gcd(r_0, r_1) = \gcd(r_1, r_2) = \ldots = \gcd(r_k, 0) = r_k.$$

Since r_k is the output of EA, this establishes our claim. □

Example 46 Compute the greatest common divisor of 1073 and 461. We apply the Euclidean algorithm and obtain the equations

$$\begin{aligned} 1073 &= 2 \cdot 461 + 151, \\ 461 &= 3 \cdot 151 + 8, \\ 151 &= 18 \cdot 8 + 7, \\ 8 &= 1 \cdot 7 + 1, \\ 7 &= 7 \cdot 1 + 0. \end{aligned}$$

Hence $1 = \gcd(1073, 461)$. Now consider 960 and 405. Then EA gives us

$$\begin{aligned} 960 &= 2 \cdot 405 + 150, \\ 405 &= 2 \cdot 150 + 105, \\ 150 &= 1 \cdot 105 + 45, \\ 105 &= 2 \cdot 45 + 15, \\ 45 &= 3 \cdot 15 + 0. \end{aligned}$$

This implies that $\gcd(960, 405) = 15$.

The following theorem, which states that the greatest common divisor of two positive natural numbers a and b can be written as a linear combination of a and b, is immediate from the Euclidian algorithm. In particular, EA can be used to compute the required factors x and y.

Theorem 47 (Bezout's identity)
If a and b are positive natural numbers, then there exist $x, y \in \mathbb{Z}$ such that

$$\gcd(a,b) \;=\; xa + yb.$$

Moreover, these integers x and y are obtained from the Euclidian algorithm EA *by eliminating the numbers $r_2, \ldots, r_{k-1}$ from the equations* (EA.3).

PROOF Among other things, EA provides us with integers $q_0, \ldots, q_{k-2}$ and $r_0, \ldots, r_k$ such that $a = r_0$, $b = r_1$, $\gcd(a,b) = r_k$, and

$$(=_n) \qquad r_{n+2} \;=\; r_n \;-\; q_n r_{n+1}.$$

for $n = 0, \ldots, k-2$. If $k = 1$, then $r_0 = q_0 \cdot r_1 + 0$ and $\gcd(a,b) = b$. Then we have

$$\gcd(a,b) \;=\; b \;=\; (-1)\cdot a \;+\; (q_0+1)\cdot b.$$

If $k > 1$, we prove by complete induction on $i \le k-2$ that

$$(*) \qquad (\exists x, y \in \mathbb{Z})(r_k \;=\; x \cdot r_{k-2-i} \;+\; y \cdot r_{k-1-i}).$$

$i = 0$. Then equation $(=_{k-2})$ yields

$$r_k \;=\; r_{k-2} \;-\; q_{k-2} \cdot r_{k-1}$$

and thus gives us (*) with $x = 1$ and $y = -q_{k-2}$.

$0 < i \le k-2$. By the induction hypothesis we then have

$$r_k \;=\; u \cdot r_{k-2-(i-1)} \;+\; v \cdot r_{k-1-(i-1)} = u \cdot r_{k-1-i} \;+\; v \cdot r_{k-i}$$

for suitable integers u and v. Together with equation $(=_{k-2-i})$ we therefore have

$$r_k \;=\; u \cdot r_{k-1-i} \;+\; v \cdot (r_{k-i-2} \;-\; q_{k-i-2} \cdot r_{k-i-1}) \;=\; v \cdot r_{k-i-2} \;+\; (u - q_{k-i-2}) \cdot r_{k-i-1}.$$

This implies (*) for $x = v$ and $y = (u - q_{k-i-2})$.

Hence we have (*) for all natural numbers i with $i \le k-2$, as claimed above. If we set $i = k-2$, then there exist $x, y \in \mathbb{Z}$ for which

$$r_k \;=\; x \cdot r_0 \;+\; y \cdot r_1.$$

Since $a = r_0$, $b = r_1$, and $\gcd(a,b) = r_k$, this is the desired identity. □

Example 48 We are interested in the greatest common divisor of 111 and 39 and in writing it as a linear combination of 111 and 39. So let us start with the EA:

$$\begin{aligned} 111 &= 2 \cdot 39 + 33, \\ 39 &= 1 \cdot 33 + 6, \\ 33 &= 5 \cdot 6 + 3, \\ 6 &= 2 \cdot 3 + 0. \end{aligned}$$

This means that $\gcd(111, 39) = 3$. By the elimination process described in the proof of Bezout's identity we obtain in addition:

$$\begin{aligned} 3 &= 1 \cdot 33 + (-5) \cdot 6 \\ &= 1 \cdot 33 + (-5) \cdot (39 - 1 \cdot 33) \\ &= (-5) \cdot 39 + 6 \cdot 33 \\ &= (-5) \cdot 39 + 6 \cdot (111 - 2 \cdot 39) \\ &= 6 \cdot 111 + (-17) \cdot 39. \end{aligned}$$

Hence 6 and −17 are the desired factors.

Corollary 49 *If p and q are different prime numbers, then we have for all integers a that*

$$p \mid a \wedge q \mid a \quad \Longrightarrow \quad pq \mid a.$$

Proof 1. Since p and q are different primes, we know that $\gcd(p, q) = 1$. Thus Bezout's identity tells us that there are $x, y \in \mathbb{Z}$ such that $1 = xp + yq$. Also, since p and q are divisors of a, there exist $u, v \in \mathbb{Z}$ satisfying $a = up$ and $a = vq$. Altogether we have

$$a = axp + ayq = vqxp + upyq = (vx + uy) \cdot pq.$$

But this equation says that a is divisible by pq. □

1.8 Partitions and equivalence relations

Partitions and equivalences are examples of further elementary but very important notions in mathematics and computer science. We present their formal definitions and several introductory examples, prove an important combinatorial principle – the so-called pigeonhole principle – and a theorem about the relationship between partitions and equivalence relations. At the end of this section we say a bit about congruences modulo n since such congruences will be taken up again in Section 1.9.

Definition 50 *A* partition *of a set S is a collection P of pairwise disjoint non-empty subsets of S such that every element of S belongs exactly to one of these subsets; this means that the following conditions are satisfied:*

(Par.1) $(\forall Q \in P)(Q \subseteq S \ \wedge \ Q \neq \varnothing)$.

(Par.2) $S = \bigcup P$.

(Par.3) $(\forall Q, R \in P)(Q \neq R \implies Q \cap R = \varnothing)$.

Example 51

1. If $S = \varnothing$, then $\varnothing$ is the only partition of S.
2. If S is the set $\{a\}$ consisting of the single object a, then $P := \{\{a\}\}$ is the only partition of S.
3. The set $\{1, 2\}$ has the following two partitions:
   - $P_1 = \{\{1\}, \{2\}\}$,
   - $P_2 = \{\{1, 2\}\}$.
4. The set $\{1, 2, 3\}$ has the following five partitions:
   - $P_1 = \{\{1\}, \{2\}, \{3\}\}$,
   - $P_2 = \{\{1, 2\}, \{3\}\}$,
   - $P_3 = \{\{1, 3\}, \{2\}\}$,
   - $P_4 = \{\{1\}, \{2, 3\}\}$,
   - $P_5 = \{\{1, 2, 3\}\}$.
5. For any set Q and non-empty proper subset R of Q, $\{R, Q \setminus R\}$ is a partition of S.

Without going into details, let us mention that the number of partitions of an n-element set ($n \in \mathbb{N}$) is the *Bell number* $\mathcal{B}_n$. As we can see from the previous example, we have $\mathcal{B}_0 = 1$, $\mathcal{B}_1 = 1$, $\mathcal{B}_2 = 2$, and $\mathcal{B}_3 = 5$. In general, the Bell numbers satisfy the following recursion:

$$\mathcal{B}_0 = 1 \quad \text{and} \quad \mathcal{B}_{n+1} = \sum_{i=0}^{n} \binom{n}{i} \mathcal{B}_i.$$

The pigeonhole principle is one of the most elementary and intuitively convincing combinatorial principles: If $n + 1$ pigeons are allocated to n pigeonholes, then at least one pigeonhole contains at least two pigeons. Here we present its original finite version and an infinite variant.

Theorem 52 (Pigeonhole principle)
Assume that we are given a set S, a positive natural number n, and a partition $\{Q_1, \ldots, Q_n\}$ of S.

1. *If S is finite and $n < |S|$, then there exists an $i \in \{1, \ldots, n\}$ such that $2 \leq |Q_i|$.*
2. *If S is infinite, then there exists an $i \in \{1, \ldots, n\}$ such that Q_i is infinite.*

PROOF We prove by complete induction on m that for all finite sets S and all partitions $\{Q_1, \ldots, Q_{m+1}\}$ of S:

$$(*) \qquad m+1 < |S| \implies (\exists i \in \mathbb{N})(1 \leq i \leq m+1 \ \wedge \ 2 \leq |Q_i|).$$

$m = 0$. Then $Q_1 = S$, and our assertion is obvious.

$m > 0$. Consider the set Q_{m+1}. If $1 < |Q_{m+1}|$, we are done. Otherwise there exists an element $a \in S$ such that $Q_{m+1} = \{a\}$. Then, for $m_0 := m - 1$, $\{Q_1, \ldots, Q_{m_0+1}\}$ is a partition of $S \setminus \{a\}$ and $m_0 + 1 < |S \setminus \{a\}|$. Hence the induction hypothesis implies what we want.

The first assertion of our theorem is immediate from (*). The second assertion follows directly from Lemma 35. □

As we will see below, the partitions of a set S are just another way of looking at equivalence relations on S in the sense that every partition of S gives rise to equivalence relations on S and vice versa.

Definition 53

1. *A binary relation R on a set S is said to be an* equivalence relation *(on S) if and only if it satisfies the following three properties:*

   (Equ.1) $(\forall x \in S)((x, x) \in R)$. *(Reflexivity)*

   (Equ.2) $(\forall x, y \in S)((x, y) \in R \implies (y, x) \in R)$. *(Symmetry)*

   (Equ.3) $(\forall x, y, z \in S)((x, y) \in R \wedge (y, z) \in R \implies (x, z) \in R)$. *(Transitivity)*

2. *Given a set S, an equivalence relation R on S, and an element $a \in S$, the* equivalence class *of a under R is defined as*

$$[a]_R := \{x \in S : (a, x) \in R\}.$$

3. *Finally, given a set S and an equivalence relation R on S, we write S/R for the collection of all equivalence classes under R, i.e.*

$$S/R := \{[a]_R : a \in S\}.$$

In the context of equivalence relations it is very common to use infix notation and to write $a\,R\,b$ instead of $(a,b) \in R$. We will do so from now on whenever it is convenient and more suggestive for the reader.

Example 54

1. Let $\triangle$ be the binary relation on $\mathbb{Z}$ defined by, for any $x, y \in \mathbb{Z}$,

$$x \triangle y \quad :\Longleftrightarrow \quad x + y \text{ is even.}$$

   Then $\triangle$ is an equivalence relation on $\mathbb{Z}$; it has two equivalence classes: the sets of the odd and the even integers.

2. Let S be the set of all students of Bern University. For $x, y \in S$ we define

$$x \triangle y \quad :\Longleftrightarrow \quad x \text{ has the same birthday as } y.$$

   Then $\triangle$ is an equivalence relation on S.

3. Let F be a function from set R to set S. For $x, y \in R$ we define

$$x \triangle y \quad :\Longleftrightarrow \quad F(x) = F(y).$$

   Then $\triangle$ is an equivalence relation on R.

The proof of the following theorem, which states the promised relationship between partitions and equivalence relations, is straightforward and left to the reader as an easy exercise.

Theorem 55 *Let S be an arbitrary set.*

1. *If P is a partition of S and $\sim_P$ is the binary relation on S defined by*

$$a \sim_P b \quad :\Longleftrightarrow \quad (\exists Q \in P)(a \in Q \wedge b \in Q)$$

   for $a, b \in S$, then $\sim_P$ is an equivalence relation on S. Also, for any $Q \in P$ and $a \in Q$ we have $Q = [a]_{\sim_P}$.

2. *If $\sim$ is an equivalence relation on S, then $S/\!\sim$ is a partition of S.*

Very important equivalence relations on $\mathbb{Z}$ are the congruences modulo n, where n is an integer greater than 0. They are the crucial concept of modular arithmetic and play and important role in many other areas of mathematics and computer science, for example in number theory, algebra, cryptography, complexity theory.

Definition 56 *Assume that $a, b, n \in \mathbb{Z}$ and $n > 0$.*

1. *a and b are called* congruent modulo *n if and only if n divides $a-b$;*

$$a \equiv_n b \quad :\Longleftrightarrow \quad a \equiv b \mod n \quad :\Longleftrightarrow \quad n \mid (a-b).$$

2. *The* residue classes modulo *n are the sets*

$$[a]_n := \{x \in \mathbb{Z} : x \equiv_n a\}.$$

3. *The set of all residue classes modulo n is denoted by $\mathbb{Z}/n\mathbb{Z}$,*

$$\mathbb{Z}/n\mathbb{Z} := \{[x]_n : x \in \mathbb{Z}\}.$$

4. *We write* $\mathrm{Rem}_n(a)$ *for the remainder on division of a by n; i.e.*

$$Rem_n(a) \in \{0, \ldots, n-1\} \quad and \quad (\exists x \in \mathbb{Z})(a = nx + \mathrm{Rem}_n(a)).$$

In the following lemma we compile a series of properties of congruences modulo n. Most of them should be know, and their proofs are straightforward so that we can omit them.

Lemma 57 *Let n be a positive natural number and a, b, c, d any integers.*

1. *$\equiv_n$ is an equivalence relation on $\mathbb{Z}$.*

2. *$\mathbb{Z}/n\mathbb{Z} = \{[0]_n, \ldots, [n-1]_n\}$.*

3. *$\mathbb{Z}/n\mathbb{Z}$ is a partition of $\mathbb{Z}$.*

4. *$a \equiv_n b \iff \mathrm{Rem}_n(a) = \mathrm{Rem}_n(b)$.*

5. *$a \equiv_n 1 \implies ab \equiv_n b$.*

6. *$a \equiv_n c \wedge b \equiv_n d \implies ab \equiv_n cd$.*

The Euclidian algorithm EA - in the form of Bezout's identity - tells us how to compute the modular multiplicative inverse of an integer modulo n.

Lemma 58 *If n is a positive natural number and a a natural number such that $1 < a < n$ and $\gcd(a, n) = 1$, then we can use the Euclidian algorithm* EA *to find a $b \in \mathbb{N}_n$ such that $ab \equiv_n 1$. This b is called the* modular multiplicative inverse *of a modulo n.*

PROOF Since $\gcd(a,n) = 1$, the EA provides us according to Bezout's identity with integers x and y such that $ax + ny = 1$. Therefore we have $ax - 1 = ny$, meaning that n divides $ax - 1$, i.e. $ax \equiv_n 1$. If $x \in \mathbb{N}_n$, we choose x as the desired integer b. Otherwise we find an integer k such that $x + kn \in \mathbb{N}_n$. Then we have

$$a(x + kn) - 1 = ax - 1 + kn.$$

But since $ax - 1$ is divisible by n, so is $ax - 1 + kn$. Hence $x + kn$ is the desired b. □

As a further "ingredient" of the RSA-algorithm, which we will discuss in the following section, we need Euler's totient function φ. The totient $\varphi(n)$ is defined for any positive natural number n as the number of natural numbers less than n that are coprime to n.

Definition 59 *For any $n \in \mathbb{Z}$, $n > 0$, we set*

$$\varphi(n) := |\{x \in \mathbb{N} : x < n \wedge \gcd(x,n) = 1\}|.$$

So we have, for example, $\varphi(1) = 1$, $\varphi(2) = 1$, $\varphi(3) = 2$, $\varphi(6) = 2$, $\varphi(9) = 6$. The above definition immediately implies that $\varphi(p) = p - 1$ for any prime number p. In general, the computation of $\varphi(n)$ is difficult. However, often it can be simplified by exploiting the multiplicativity property of φ, see below.

Euler's totient function is an important tool in number theory, but time does not permit to go into details. It is treated in most textbooks on number theory; see, for example, Niven, Zuckerman, and Montgomery [8]. We confine ourselves to stating a few important properties which will be used later.

Theorem 60 *Let p and q be different prime numbers, m and n positive natural numbers, and a an arbitrary integer. Then we have:*

1. $\varphi(p) = p - 1$.
2. $\gcd(m,n) = 1 \implies \varphi(m \cdot n) = \varphi(m) \cdot \varphi(n)$.
3. $\varphi(p \cdot q) = (p-1)(q-1)$.
4. *Euler:* $\gcd(a,n) = 1 \implies a^{\varphi(n)} \equiv_n 1$.
5. *Fermat's little theorem:* $p \nmid a \implies a^{p-1} \equiv_p 1$.

The first assertion is obvious since $\gcd(x,p) = 1$ for every element x of $\{1,\ldots,p-1\}$. The third assertion is immediate from the first and the second, and the fifth follows directly from the first and the fourth. For the proofs of the second and the fourth assertion please consult the literature.

1.9 The RSA-algorithm

The RSA-algorithm – named after R. Rivest, A. Shamir, and L. Adelman – is one of the most famous algorithms for public-key cryptography; see [9]. The underlying principle of this algorithm is that the multiplication of large numbers is simple whereas the factorization of large numbers – according to our present knowledge – is unfeasible in reasonable time.

The RSA-algorithm makes use of a *public key* and a *private key* and involves the following three steps:

- generation of the public key and the private key,
- encryption,
- decryption.

For the presentation of the RSA-cryptosystem, it is convenient to introduce, for any positive natural numbers n and a, the following auxiliary function $\Theta_{(n,a)}$,

$$\Theta_{(n,a)} : \mathbb{N}_n \to \mathbb{N}_n; \quad \Theta_{(n,a)}(x) := \mathrm{Rem}_n(x^a).$$

Generation of the keys

(G1) We choose two different large prime numbers p and q, which we keep secret.

(G2) We compute $n = p\,q$.

(G3) We compute $\varphi(n)$. This is easily done since we know that $n = p\,q$ and therefore, because of Theorem 60, $\varphi(n) = (p-1)(q-1)$.

(G4) We choose an integer e such that $1 < e < \varphi(n)$ and $\gcd(e, \varphi(n)) = 1$. The pair (n, e) is the **public key**. It may be released freely.

(G5) Now we employ Lemma 58 to determine a d such that $ed \equiv_{\varphi(n)} 1$. The pair (n, d) is the **private key**. It has to be kept secret.

Encryption

(E1) We transmit the public key (n, e) to our colleague $\mathcal{C}$ who wants to send us a message $\mathcal{M}$.

(E2) $\mathcal{C}$ now translates the message $\mathcal{M}$ by some standard, agreed-upon, and reversible protocol into an integer m with $1 < m < n$.

(E3) Then $\mathcal{C}$ computes $c = \Theta_{(n,e)}(m)$ and transmits c to us. He may use an open channel for this transmission.

Decryption

(D1) After having received c, we simply apply the function $\Theta_{(n,d)}$ to c and obtain the value $\Theta_{(n,d)}(c)$. According to Theorem 62 below, we know that $\Theta_{(n,d)}(c) = m$.

(D2) After knowing m, the original message $\mathcal{M}$ is reconstructed by reversing the protocol used in (E2).

Example 61 We pick the prime numbers $p = 53$ and $q = 61$. Then $n = 3233$ and $\varphi(n) = 3120$. For generating the public key we choose, in addition, $e = 1013$. e is a prime number, hence $\gcd(e, \varphi(n)) = 1$. The public key is $(3233, 1013)$.

Applying the Euclidian algorithm EA as it has to be used for the proof of Lemma 58 (please do it in detail!), we obtain

$$e \cdot 77 \; + \; 3120 \cdot (-25) \; = \; 1.$$

Hence $(3233, 77)$ is the private key. Now let us suppose that the message $\mathcal{M}$ is coded as the number 10. Then

$$\Theta_{(3233,1013)}(10) \; = \; \mathrm{Rem}_{3233}(10^{1013}) \; = \; 2189,$$

and the number 2189 will be transmitted. For decrypting, we proceed as follows:

$$\Theta_{(3233,77)}(2189) \; = \; \mathrm{Rem}_{3233}(2189^{77}) \; = \; 10.$$

Theorem 62 *Assume that we are given two different prime numbers p and q, the product $n = pq$, the public key (n, e), and the corresponding private key (n, d), all computed as described above. Then we have for all $m \in \mathbb{N}_n$:*

1. $(m^e)^d \equiv_n m$.
2. $\Theta_{(n,d)}(\Theta_{(n,e)}(m)) = m$.

Proof 1. By Corollary 49 it is sufficient to show $(m^e)^d \equiv_p m$ and $(m^e)^d \equiv_q m$, and by symmetry we can confine us to do it for p. Now we distingush the following two cases:

(i) $p \mid m$. Because of $ed \equiv_{\varphi(n)} 1$ we have $ed \geq 1$, hence p is also a divisor of $(m^e)^d$. Therefore $(m^e)^d \equiv_p m$.

(ii) $p \nmid m$. Then the little Fermat implies

$$(*) \qquad m^{p-1} \equiv_p 1.$$

Because of $ed \equiv_{\varphi(n)} 1$ we know that there exists an $x \in \mathbb{N}$ such that $ed = 1 + x\varphi(n)$. This implies by Theorem 60.3

$$(m^e)^d = m^{1+x\varphi(n)} = m^{1+x(p-1)(q-1)} = m \cdot (m^{p-1})^{x(q-1)}.$$

In view of Lemma 57.5, Lemma 57.6, and (*) we can conclude from this equation that $(m^e)^d \equiv_p m$.

2. Now we set $k := \Theta_{(n,e)}(m) = \mathrm{Rem}_n(m^e)$. By Lemma 57.4 we thus have $k \equiv_n m^e$, and Lemma 57.6 implies

$$(**) \qquad k^d \equiv_n (m^e)^d.$$

On the other hand, we also have, making use of Lemma 57.4 again,

$$\Theta_{(n,d)}(\Theta_{(n,e)}(m)) = \Theta_{(n,d)}(k) = \mathrm{Rem}_n(k^d) \equiv_n k^d.$$

Together with (**) and the first assertion of this theorem we obtain

$$\Theta_{(n,d)}(\Theta_{(n,e)}(m)) \equiv_n m.$$

But $\Theta_{(n,d)}(\Theta_{(n,e)}(m))$ and m are elements of $\mathbb{N}_n$, and so this equivalence implies the desired equality. □

Discussion

- Finding large prime numbers of adequate size can be done by testing random numbers with probabilistic primality tests to quickly eliminate all non-primes.
- Once the public key (n, e) is available, there are quick methods to compute the number $c = \Theta_{(n,e)}(m)$.
- Accordingly, once the private key (n, d) is available, there are quick methods to compute the number $m = \Theta_{(n,d)}(c)$.
- The problem of the RSA-algorithm is to find, for the given c, a value m such that $c \equiv_n m^e$.
- According to our present knowledge, the most promising approach to solving this problem is to compute d from (n, e). This is easy, once n has been factored into p and q. Then $\varphi(n)$ is known to be $(p-1)(q-1)$, and then the computation of d from e is by (consequences of) the Euclidian algorithm EA, as described above. Actually, Rivest, Shamir, and Adelman [9] contains a proof that computing d from (n, e) is as hard (complicated) as factoring n.

- So far, no feasible method for factoring large natural numbers into primes has been found. However, it has not been proved that such a method does not exist.
- Also, it has not been proved – although it is considered highly unlikely – that there is no alternative method for cracking the RSA-cryptosystem which avoids factoring.

1.10 Graphs

Graphs are structures which model the relationships between objects from a given domain of interest. They consist of *vertices* – sometimes also called *nodes* – and *edges* which connect pairs of vertices. In general, such vertices may be undirected or directed, and it is also possible to associate weights to the edges. In this section we confine us to presenting some very basic definitions and two famous problems. In the following section we then turn to very specific graphs, so-called trees

More on graphs and trees and, in particular, on graph and tree algorithms is taught in the course *Datenstrukturen und Algorithmen*. There are also numerous textbooks on graph theory, Diestel [4] being one excellent example. Much of what will be treated in this section and the following section is taken from the lecture notes of Buchholz [1] and Schwichtenberg [12].

Definition 63

1. *An undirected* graph *is a pair* $G = (V, E)$ *consisting of a non-empty set* V *of* vertices *and a set* $E \subseteq \{X \subseteq V : |X| = 2\}$. *If* $\{a, b\} \in E$, *we say that the edge* $\{a, b\}$ connects *the vertices* a *and* b.
2. *An undirected* graph $G = (V, E)$ *is called* finite *if* V *is a finite set.*
3. *A weighted (undirected) graph is a triple* $G = (V, E, W)$ *such that* (V, E) *is an undirected graph and* W *a function from* E *to* $\mathbb{R}^+$.

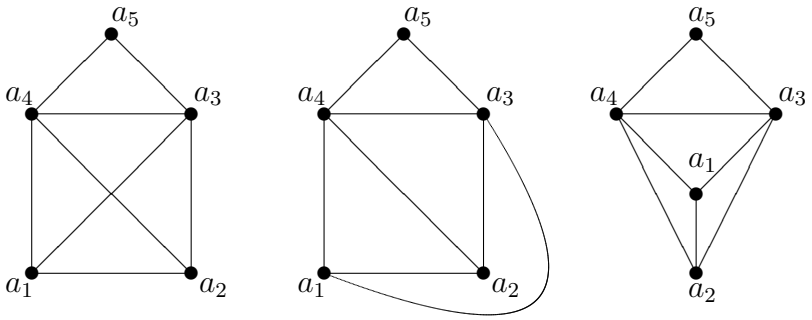
Observe that our definition does not permit loops, i.e. edges which go from a vertex to itself. Also, directed graphs will not be considered in these lectures and, therefore, from now on graphs will be identified with undirected graphs.

Graphs are often depicted as diagrams.The graph $G = (V, E)$ with

$$V = \{a_1, a_2, a_3, a_4, a_5\},$$

$$E = \{\{a_1, a_2\}, \{a_1, a_3\}, \{a_1, a_4\}, \{a_2, a_3\}, \{a_2, a_4\}, \{a_3, a_4\}, \{a_3, a_5\}, \{a_4, a_5\}\}$$

can be drawn, for example, in one of the following three ways:



An alternative way to represent a finite graph is via its adjacency matrix. This is the typical way, computer science deals with finite graphs.

Definition 64 *Let $G = (V, E)$ be a finite graph with $V = \{a_1, \ldots, a_n\}$. Then the* adjacency matrix *M_G of G is the $n \times n$ matrix*

$$\begin{pmatrix} m_{11} & \cdots & m_{1n} \\ \vdots & \ddots & \vdots \\ m_{n1} & \cdots & m_{nn} \end{pmatrix}$$

where, for any $i, j \in \{1, \ldots, n\}$,

$$m_{ij} := \begin{cases} 1 & \text{if } \{a_i, a_j\} \in E, \\ 0 & \text{if } \{a_i, a_j\} \notin E. \end{cases}$$

Obviously, the adjacency matrix of any (undirected) graph is symmetric. The adjacency matrix of the graph depicted above is the following matrix:

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

Definition 65 *For every vertex a of a graph $G = (V, E)$ we set*

$$\deg_G(a) := |\{e \in E : a \in e\}|.$$

$\deg_G(a)$ *is called the* degree *of a with respect to G and is the number of edges which contain a.*

The following lemma states a simple but important property of finite graphs which nicely relates the degrees of their vertices to the number of edges. As an immediate consequence of this lemma we obtain that the number of vertices which have an odd degree is even. This is the reason why it is called handshake lemma: *The number of persons at a party, who shake hands with an odd number of guests, is even.*

Lemma 66 (Handshake lemma)
For every finite graph $G = (V, E)$ we have

$$\sum_{a \in V} deg_G(a) \;=\; 2 \cdot |E|.$$

PROOF For any $a \in V$ and $e \in E$ we set

$$\mathrm{char}(a, e) \;:=\; \begin{cases} 1 & \text{if } a \in e, \\ 0 & \text{if } a \notin e \end{cases}$$

and obtain

$$\sum_{a \in V} \deg_G(a) \;=\; \sum_{a \in V} \sum_{e \in E} \mathrm{char}(a, e) \;=\; \sum_{e \in E} \sum_{a \in V} \mathrm{char}(a, e) \;=\; \sum_{e \in E} 2 \;=\; 2 \cdot |E|.$$

If we add up the degrees of all vertices, every edge $\{a, b\}$ is counted twice, once in connection with $\deg_G(a)$ and once in connection with $\deg_G(b)$. □

Definition 67 *Assume that we are given a graph $G = (V, E)$.*

1. *A* path *in G is a finite sequence $(a_0 \ldots, a_n)$ of vertices such that $\{a_i, a_{i+1}\} \in E$ for all $i = 0, \ldots, n-1$. The* length *of a path is the number of its edges; i.e. a path $(a_0 \ldots, a_n)$ has length n.*

2. *A path $(a_0 \ldots, a_n)$ in G is called* closed *if $a_0 = a_n$; otherwise it is called* open.

3. *A* trail *in G is a path $(a_0, \ldots, a_n)$ without repeating edges, i.e. $\{a_i, a_{i+1}\} \neq \{a_j, a_{j+1}\}$ for all $0 \leq i < j \leq n-1$. A trail $(a_0, \ldots, a_n)$ is called a trail from a_0 to a_n.*

4. *A closed trail of length ≥ 3 is called a* cycle.

5. *Two nodes a and b are called* connected *in G if and only if there exists a path $(a_0 \ldots, a_n)$ in G such that $a = a_0$ and $b = a_n$.*

6. *For any $a \in V$ we set $[a]_G := \{b \in V : a \text{ and } b \text{ are connected}\}$. These sets are called the* components *of G.*

7. *We write $\mathrm{Comp}(G)$ for the set of all components of G; i.e.*

$$\mathrm{Comp}(G) \;:=\; \{[a]_G : a \in V\}.$$

8. *G is called* connected *if and only if any two $a, b \in E$ are connected. Equivalently, we could say that $|\mathrm{Comp}(G)| = 1$.*

On any graph $G = (V, E)$, the relation "vertex a and vertex b are connected in G" is obviously an equivalence relation on G. According to what we have seen in Theorem 55, $\mathrm{Comp}(G)$ is therefore a partition of V.

Lemma 68 *If two different vertices a and b of the graph G are connected in G, then there exists a trail in G from a to b.*

PROOF By $<$-induction we prove the following auxiliary assertion for all natural numbers n:

> If a and b are different and connected in G by a path of length n, then there exists a trail in G from a to b.

So assume that there exists a path $(a_0, \ldots, a_n)$ in G such that $a_0 = a$ and $a_n = b$. If this path is a trail, we are done. Otherwise there exist $i, j \in \{0, \ldots, n\}$ such that $i < j$ and $a_i = a_j$. Because of $a_0 \neq a_n$ this implies $0 < i$ or $j < n$. Without loss of generality we assume $0 < i$. Then $(a_0, \ldots, a_{i-1}, a_j, \ldots, a_n)$ is a path in G from a to b of length less than n. By the induction hypothesis we therefore know that there exists a trail in G from a to b.

This finishes the proof of our auxiliary assertion which, in turn, immediately implies our lemma. □

The following lemma provides an alternative characterization of the connectedness of a graph and says something about the number of components of a slightly reduced graph. Its proof is left to the reader as an exercise.

Lemma 69 *Let $G = (V, E)$ be a graph.*

1. *G is connected if and only if we have for any partition $\{V_1, V_2\}$ of V that*

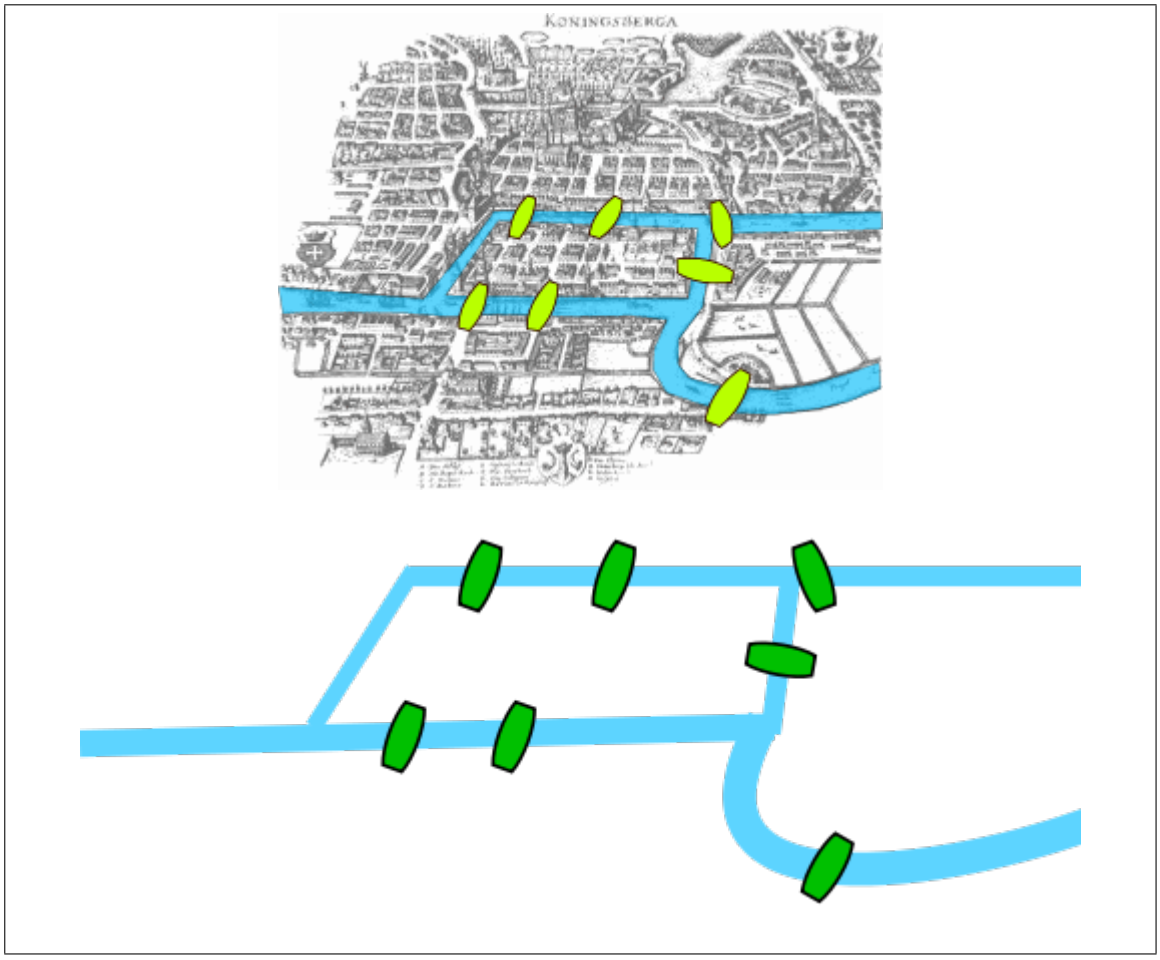
$$(\exists a \in V_1)(\exists b \in V_2)(\{a, b\} \in E).$$

2. *If $e \in E$ and $G' = (V, E \setminus \{e\})$, then we have:*
   (a) $|\mathrm{Comp}(G')| = |\mathrm{Comp}(G)|$ *or* $|\mathrm{Comp}(G')| = |\mathrm{Comp}(G)| + 1$.
   (b) $|\mathrm{Comp}(G')| = |\mathrm{Comp}(G)|$ *if and only if* $\mathrm{Comp}(G') = \mathrm{Comp}(G)$ *if and only if G has a cycle which contains e.*

Before continuing with the next concept, let us look at the following well-known problem, quoted, in this formulation, from Wikipedia.

Königsberger Brückenproblem

Das Königsberger Brückenproblem ist eine mathematische Fragestellung des frühen 18. Jahrhunderts, die anhand von sieben Brücken der Stadt Königsberg illustriert wurde. Das Problem bestand darin, zu klären, ob es einen Weg gibt, bei dem man alle sieben Brücken über den Pregel genau einmal überquert, und wenn ja, ob auch ein Rundweg möglich ist, bei dem man wieder zum Ausgangspunkt gelangt. Wie Leonhard Euler 1736 bewies, war ein solcher Weg bzw. "Eulerscher Weg" in Königsberg nicht möglich, da zu allen vier Ufergebieten bzw. Inseln eine ungerade Zahl von Brücken führte. Es dürfte maximal zwei Ufer (Knoten) mit einer ungeraden Zahl von angeschlossenen Brücken (Kanten) geben. Diese zwei Ufer könnten Ausgangs- bzw. Endpunkt sein. Die restlichen Ufer müssten eine gerade Anzahl von Brücken haben, um sie auch wieder verlassen zu können.



Das Brückenproblem ist kein klassisches geometrisches Problem, da es nicht auf die präzise Lage der Brücken ankommt, sondern nur darauf, welche Brücke welche

Inseln miteinander verbindet. Es handelt sich deshalb um ein topologisches Problem, das Euler mit Methoden löste, die wir heute der Graphentheorie zurechnen. Das Problem lässt sich auf beliebige Graphen verallgemeinern, und auf die Frage, ob es darin einen Zyklus gibt, der alle Kanten genau einmal benutzt. Ein solcher Zyklus wird als Eulerkreis bezeichnet und ein Graph, der einen Eulerkreis besitzt, als eulersch. Die Frage, ob ein Graph eulersch ist, lässt sich relativ einfach beantworten und ist auch in gerichteten Graphen und Graphen mit Mehrfachkanten möglich.

Definition 70 *Assume that we are given a graph $G = (V, E)$.*

1. *A trail $(a_0 \ldots, a_n)$ in G is called* Eulerian *if and only if every edge occurs on it, i.e. for any $e \in E$ there exists an $i \in \{0, \ldots, n-1\}$ such that $e = \{a_i, a_{i+1}\}$.*

2. *An* Eulerian cycle *is a closed Eulerian trail. G is called* Eulerian *if and only if it has an Eulerian cycle.*

Our next aim is to come up with a necessary and sufficient condition for a finite connected graph to be Eulerian. To achieve this, we need two preparatory lemmas.

Lemma 71 *Let $\mathcal{T} = (a_0, \ldots, a_n)$ be a trail in the graph $G = (V, E)$. For any $a \in V$ we set*

$$\deg_{\mathcal{T}}(a) := |\{i < n : a \in \{a_i, a_{i+1}\}\}|.$$

Then we have for all $a \in V$ that

$$\deg_{\mathcal{T}}(a) \equiv_2 \begin{cases} 1 & \text{if } a_0 \neq a_n \wedge (a = a_0 \vee a = a_n), \\ 0 & \text{otherwise.} \end{cases}$$

Proof For $k := |\{i : 0 < i < n \wedge a = a_i\}|$ we have for any $a \in V$ that

$$\deg_{\mathcal{T}}(a) = \begin{cases} 2k+2 & \text{if } a = a_0 = a_n, \\ 2k+1 & \text{if } a_0 \neq a_n \wedge (a = a_0 \vee a = a_n), \\ 2k & \text{if } a \neq a_0 \wedge a \neq a_n. \end{cases}$$

Our assertion follows directly from this equation. □

Lemma 72 *Let $G = (V, E)$ be a finite connected graph such that all its vertices have even degree. In addition, assume that $(a_0, \ldots, a_n)$ is a closed trail in G which does not contain all edges of G. Then we can find a cycle $\mathcal{T}$ which extends $(a_0, \ldots, a_n)$ properly.*

PROOF We first convince ourselves that we can pick an edge $\{b_0, b_1\}$ not contained in $(a_0, \ldots, a_n)$ but with $b_0 \in \{a_0, \ldots, a_n\}$:

Let $\{c, d\}$ be an edge not on $(a_0, \ldots, a_n)$. If $c \in \{a_0, \ldots, a_n\}$, then set $b_0 := c$ and $b_1 := d$. If $c \notin \{a_0, \ldots, a_n\}$, then, since G is connected, there exists a path from c to an element of $\{a_0, \ldots, a_n\}$. This path contains an edge of the form we need.

Following this principle, we now successively pick edges $b_2, \ldots, b_i, \ldots$ such that $(b_0, \ldots, b_i)$ is a trail in G not containing edges from $(a_0, \ldots, a_n)$. However, G is finite and therefore we will end up with a trail $(b_0, \ldots, b_k)$ such that all edges starting from b_k belong to $(a_0, \ldots, a_n)$ or $(b_0, \ldots, b_k)$. Therefore we have

$$\deg_G(b_k) \;=\; \deg_{(a_0,\ldots,a_n)}(b_k) + \deg_{(b_0,\ldots,b_k)}(b_k).$$

From the previous lemma we deduce that $\deg_{(a_0,\ldots,a_n)}(b_k)$ is even. By assumption, all vertices of G have even degree, hence $\deg_{(b_0,\ldots,b_k)}(b_k)$ has to be even as well. Now we apply the previous lemma once more and obtain $b_0 = b_k$.

We also know that $a_0 = a_n$ and $b_0 = a_j$ for some $j \leq n$. Hence $a_j = b_0 = b_k$ and

$$(a_j, a_{j+1}, \ldots, a_n, a_1, \ldots, a_j, b_1, \ldots, b_k)$$

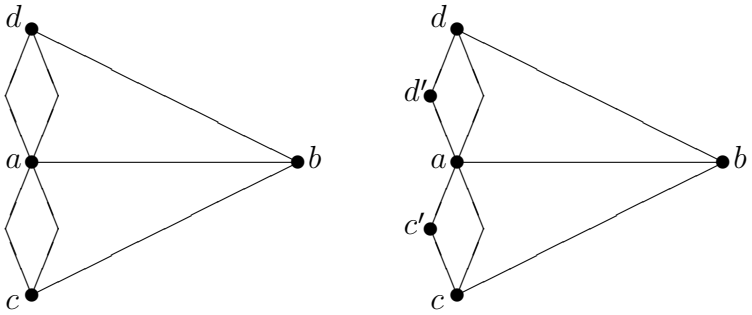
is the desired cycle which extends $(a_0, \ldots, a_n)$ properly. □

Theorem 73 *Let G be a finite connected graph. Then G is Eulerian if and only if all its vertices have even degree.*

PROOF If G is Eulerian, then G has an Eulerian cycle $\mathcal{T}$. For every vertex a we thus have $\deg_G(a) = \deg_{\mathcal{T}}(a)$. Moreover, according to Lemma 71, $\deg_{\mathcal{T}}(a) \equiv_2 0$, hence $\deg_G(a)$ is even.

If the degrees of all vertices of G are even, we simply have to iterate the previous lemma to obtain an Eulerian cycle. □

Now let us come back to the Königsberger Brückenproblem. If we recall the abstract representation of the situation we see that it is not a graph but a multi-graph (several edges between two vertices). However, it can be easily transformed into a graph by introducing auxiliary vertices:



The Königsberger Brückenproblem is solvable if and only if this is and Eulerian graph. According to our previous theorem, however, this is not the case.

We end this section by briefly discussing the famous *traveling salesperson problem*: Given a list of cities and their pairwise distances, the task is to find a shortest possible tour that visits each city exactly once.

The traveling salesperson problem can be easily reformulated in terms of weighted graphs. The cities become the vertices, we have an edge between two vertices, provided that the corresponding cities are directly connected, and the weight function assigns the distance between two directly connected cities to the corresponding edge.

The problem is known to be NP-hard (for an exact explanation of what that means, attend the course *Berechenbarkeit und Komplexität*), and all known approaches to finding optimal solutions are known to be exponential in the number of cities. It is one of the most famous problems in computer science to determine whether or not a polynomial solution exists.

1.11 Trees

Trees are specific graphs which are of great independent interest. In computer science, trees play an important role in connection with, for example, hierarchical data structures and searching (search trees).

Definition 74

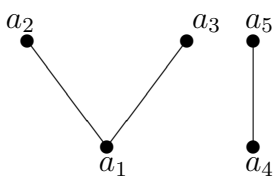
1. *A graph without cycles is called* acyclic.
2. *A* tree *is a connected acyclic graph.*

Example 75

1. The graph $G_1 = (V_1, E_1)$ with

$$V_1 = \{a_1, a_2, a_3, a_4, a_5\},$$
$$E_1 = \{\{a_1, a_2\}, \{a_1, a_3\}, \{a_4, a_5\}\}$$

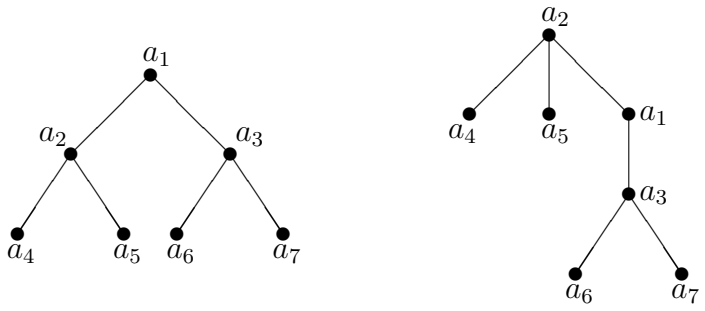
   is acyclic but not a tree since it is not connected.



2. On the other hand, the graph $G_2 = (V_2, E_2)$ with

$$V_2 = \{a_1, a_2, a_3, a_4, a_5, a_6, a_7\},$$
$$E_2 = \{\{a_1, a_2\}, \{a_1, a_3\}, \{a_2, a_4\}, \{a_2, a_5\}, \{a_3, a_6\}, \{a_3, a_7\}\}$$

is a tree. It can be drawn, for example, as follows:



There exists a simple but very useful relationship between the number of vertices and edges of a finite graph, taking its number of components into account.

Lemma 76 *For any finite graph $G = (V, E)$ we have:*

1. $|V| = |E| + |\mathrm{Comp}(G)|$ *if G is acyclic.*
2. $|V| < |E| + |\mathrm{Comp}(G)|$ *if G has a cycle.*

PROOF We show these two assertion simultaneously by complete induction on $|E|$.

$|E| = 0$. Then G is acyclic, and we have $\mathrm{Comp}(G) = \{\{a\} : a \in V\}$. This implies $|V| = |E| + |\mathrm{Comp}(G)|$.

$|E| > 0$. If G has a cycle, then we pick an edge e of his cycle and consider the reduced graph $G' := (V, E \setminus \{e\})$. By the induction hypothesis we thus have

$$|V| \leq |E \setminus \{e\}| + |\mathrm{Comp}(G')| = (|E| - 1) + |\mathrm{Comp}(G')|.$$

Furthermore, Lemma 69.2 tells us that $\mathrm{Comp}(G') = \mathrm{Comp}(G)$, and so we conclude

$$|V| \leq (|E| - 1) + |\mathrm{Comp}(G)| < |E| + |\mathrm{Comp}(G)|.$$

If G is acyclic, we pick some $e \in E$ and set again $G' := (V, E \setminus \{e\})$. Then G' is acyclic as well, and the induction hypothesis yields

$$|V| = |E \setminus \{e\}| + |\mathrm{Comp}(G')| = (|E| - 1) + |\mathrm{Comp}(G')|.$$

But because of Lemma 69.2 we know know that $|\mathrm{Comp}(G')| = |\mathrm{Comp}(G)| + 1$, hence

$$|V| \;=\; (|E| - 1) + (|\mathrm{Comp}(G)| + 1) \;=\; |E| + |\mathrm{Comp}(G)|.$$

This finishes our proof. □

This lemma provides a crucial step in establishing the following characterization of finite acyclic graphs.

Theorem 77 *Let $G = (V, E)$ be a finite graph. Then the following three assertions are equivalent:*

(1) G is acyclic.

(2) Given two different vertices a and b of G, there is at most one trail in G which goes from a to b.

(3) $|V| = |E| + |\mathrm{Comp}(G)|$.

Proof From the previous lemma we immediately conclude that assertion (1) is equivalent to assertion (3).

(1) $\Longrightarrow$ (2). Assume that a and b are different vertices of G and $(a_0, \ldots, a_m)$ and $(b_0, \ldots, b_n)$ two different trails from a to b. Without loss of generality we also assume that $m \leq n$.

If $a_i = b_i$ for all $i = 0, \ldots, m$, then $m < n$, $b_m = b = b_n$, and $(b_m, \ldots, b_n)$ is a cycle.

If $a_i \neq b_i$ for some $i = 0, \ldots, m$, then pick the least $r < m$ such that $a_r \neq b_r$. We also know that $b_n = b = a_m \in \{a_0, \ldots, a_m\}$. Hence there exists a least $s > r$ with $b_s \in \{a_0, \ldots, a_m\}$, say $b_s = a_t$. If $r \leq t$, then $(b_{r-1}, \ldots, b_s, a_{t-1}, \ldots, a_{r-1})$ is a cycle; if $t < r$, then $(b_t, \ldots, b_r, \ldots, b_s)$ is a cycle.

Always there is a contradiction to the assumption that G is acyclic.

(2) $\Longrightarrow$ (1). If $(a_0, \ldots, a_n)$ were cycle in G, then (a_0, a_1) and $(a_n, \ldots, a_1)$ were two different trails from a_0 to a_1, violating assumption (2). □

Definition 78 *Let $G = (V, E)$ be a graph.*

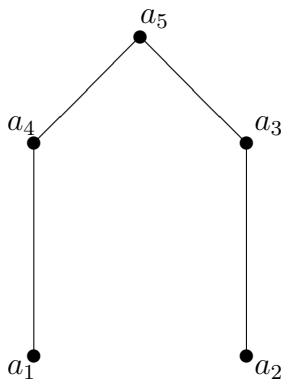
1. *Any graph $G' = (V', E')$ with $V' \subseteq V$ and $E' \subseteq E$ is called a* subgraph *of G.*
2. *A* spanning forest *of G is an acyclic subgraph G' of G for which* $\mathrm{Comp}(G') = \mathrm{Comp}(G)$.
3. *A spanning forest of G which is a tree is called a* spanning tree *of G.*

Example 79 Let us look again at the graph $G = (V, E)$ with

$$V = \{a_1, a_2, a_3, a_4, a_5\},$$

$$E = \{\{a_1, a_2\}, \{a_1, a_3\}, \{a_1, a_4\}, \{a_2, a_3\}, \{a_2, a_4\}, \{a_3, a_4\}, \{a_3, a_5\}, \{a_4, a_5\}\}.$$

Then $G' = (V, E')$ with $E' := \{\{a_1, a_4\}, \{a_4, a_5\}, \{a_5, a_3\}, \{a_3, a_2\}\}$ is a spanning tree of G, but there exist many others.



Before turning to the famous Kruskal's algorithm which computes the minimum spanning tree in a weighted connected graph, some preliminary observations.

Lemma 80 *If G' is a spanning forest of a finite acyclic graph G, then $G' = G$.*

PROOF Assume that $G = (V, E)$ and $G' = (V', E')$. By assumption we then have $E' \subseteq E$ and $\mathrm{Comp}(G') = \mathrm{Comp}(G)$. In view of Theorem 77 we can further conclude that

$$|E'| = |V| - |\mathrm{Comp}(G')| = |V| - |\mathrm{Comp}(G)| = |E|.$$

However, this immediately implies that $E' = E$, and thus $G' = G$. □

Theorem 81 *Let $G = (V, E)$ be a finite graph and (V, E_0) an acyclic subgraph of G. Then there exists an $E_1 \supseteq E_0$ such that (V, E_1) is a spanning forest of G.*

PROOF We show this assertion by <-induction on $|E|$. If G is acyclic, we simply set $E_1 := E$, and we are done. If G is not acyclic, then there exists a cycle in G. On the other hand, (V, E_0) is acyclic, and so there has to be an edge e on this cycle which does not belong to E_0. This means that $E_0 \subseteq E \setminus \{e\}$, and we set $G' := (V, E \setminus \{e\})$. By the induction hypothesis there exists an $E_1 \supseteq E_0$ such that (V, E_1) is a spanning forest of G'. Because of Lemma 69.2 we also know that $\mathrm{Comp}(G') = \mathrm{Comp}(G)$, hence G' is a spanning forest of G as well. □

Corollary 82 *Every finite connected graph has a spanning tree.*

Lemma 83 *Let $G = (V, E)$ and $G' = (V, E')$ be finite acyclic graphs and assume that $|E| < |E'|$. Then there exists an edge $e \in E' \setminus E$ such that also $(V, E \cup \{e\})$ is acyclic.*

PROOF Let $V_1, \ldots, V_n$ be the components of G. Now we make use of Lemma 76 and the assumption $|E| < |E'|$ in order to conclude that G' has at most $n-1$ components. Hence there exist different numbers $i, j \in \{1, \ldots, n\}$ and an edge $e = \{a, b\} \in E' \setminus E$ such that $a \in V_i$ and $b \in V_j$. Clearly, $(V, E \cup \{e\})$ is acyclic. □

Lemma 84 *Let $G = (V, E)$ be a finite graph and* Acg *the set of all subsets X of E such that (V, X) is an acyclic graph. Then we have:*

$$(V, F) \text{ is a spanning forest of } G \iff F \text{ is maximal in } (\mathrm{Acg}, \subseteq).$$

PROOF Direction from right to left: Let F be maximal in $(\mathrm{Acg}, \subseteq)$. From Theorem 81 we know that there exists an $F' \supseteq F$ such that (V, F') is a spanning forest of G. Hence $F' \in \mathrm{Acg}$ as well, and the maximality of F implies $F = F'$.

Direction from left to right: Let (V, F) be a spanning forest of G. Then we have $F \in \mathrm{Acg}$, and there exists a maximal $F' \in \mathrm{Acg}$ with $F \subseteq F'$. Because of the direction from right to left, $G' := (V, F')$ is a spanning forest of G. Now we apply Theorem 77 and conclude

$$|F| = |V| - |\mathrm{Comp}(G)| = |V| - |\mathrm{Comp}(G')| = |F'|.$$

Consequently, F is identical to F' and therefore maximal in $(\mathrm{Acg}, \subseteq)$. □

Definition 85 *Let $G = (V, E, W)$ be a weighted graph and F a finite subset of E. Then we set*

$$W(F) := \sum_{e \in F} W(e)$$

and call $W(F)$ the weight *of F.*

Theorem 86 (Kruskal's algorithm)

Let $G = (V, E, W)$ be a weighted graph such that $E = \{e_1, \ldots, e_m\}$ with pairwise different edges $e_1, \ldots, e_m$ and

$$W(e_1) \leq \ldots \leq W(e_m).$$

Now we set for $n = 0, \ldots, m-1$:

$$F_0 := \varnothing \quad \text{and} \quad F_{n+1} := \begin{cases} F_n \cup \{e_{n+1}\} & \text{if } (V, F_n \cup \{e_{n+1}\}) \text{ is acyclic,} \\ F_n & \text{otherwise.} \end{cases}$$

Then (V, F_m) is a W-minimal spanning forest of G; i.e. (V, F_m) is a spanning forest of G, and $W(F_m) \leq W(F)$ for any spanning forest (V, F) of G.

PROOF It is clear that F_m is a maximal acyclic subset of E. Hence the previous lemma tells us that (V, F_m) is a spanning forest of G.

We also know that F_m can be written as $\{e_{\ell(1)}, \ldots, e_{\ell(k)}\}$ with $\ell(1) < \ldots < \ell(k)$. Now choose a spanning forest $G' := (V, F)$ of G. By Theorem 77 we then have

$$|F| = |V| - |\mathrm{Comp}((V,F))| = |V| - |\mathrm{Comp}(G)| = |V| - |\mathrm{Comp}((V,F_m))| = |F_m| = k.$$

Therefore we can write F as $\{f_1, \ldots, f_k\}$ with $W(f_1) \leq \ldots \leq W(f_k)$.

Assumption: $W(F) < W(F_m)$.

Then there exists an $i \in \{1, \ldots, k\}$ such that $W(f_i) < W(e_{\ell(i)})$ and $W(f_j) \geq W(e_{\ell(j)})$ for $j = 1, \ldots, i-1$. Now consider

$$G_0 := \{e_{\ell(1)}, \ldots, e_{\ell(i-1)}\} \quad \text{and} \quad G_1 := \{f_1, \ldots, f_i\}$$

and apply Lemma 83 to G_0 and G_1. It gives us an $r \in \{1, \ldots, i\}$ for which $f_r \notin G_0$ and $(V, G_0 \cup \{f_r\})$ is a forest. We also know

$$W(f_r) \leq W(f_i) < W(e_{\ell(i)}),$$

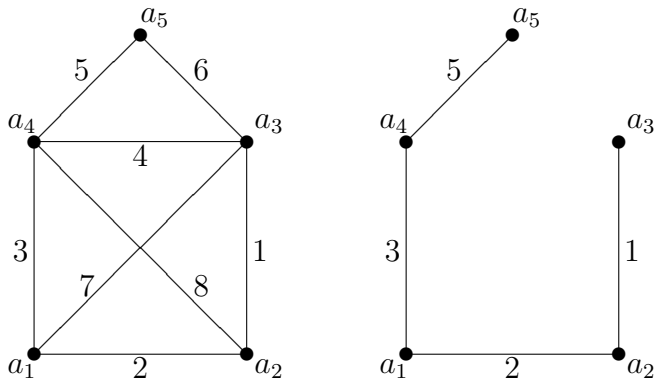
and therefore $f_r = e_s$ for some $s < \ell(i)$. By the definition of the sets F_n, $0 \leq n \leq m$, we have

$$F_{s-1} \subseteq F_s \subseteq F_{\ell(i-1)} = \{e_{\ell(1)}, \ldots, e_{\ell(i-1)}\} = G_0.$$

$(V, G_0 \cup \{f_r\})$ is a forest, hence $(V, F_{s-1} \cup \{e_s\})$ is a forest as well, and thus $F_s = F_{s-1} \cup \{e_s\}$. This implies $f_r = e_s \in F_s \subseteq G_0$.

This is a contradiction to our original choice of f_r; so $W(F_m) \leq W(F)$, and the theorem is proved. □

Example 87 Again we take our graph from Example 79, assigns some weights to its edges. Then Kruskal's algorithm yields a spanning tree.



The next and final part of this section deals with possibly infinite trees. Moreover, now we restrict our attention to trees in which one special vertex is singled out as root.

Definition 88

1. *A* rooted tree *is a triple* $T = (V, E, r)$ *such that* (V, E) *is a tree and* $r \in V$*; the vertex* r *is called the* root *of* T*.*

2. *Given a rooted tree* $T = (V, E, r)$*, the tree-order is the partial ordering on the vertices of a tree with* $a \sqsubseteq b$ *if and only if the unique path from the root* r *to* b *passes through* a*.*

3. *In a rooted tree* $T = (V, E, r)$*, the* parent *of a vertex is the vertex connected to it on the path to the root; every vertex except the root has a unique parent. A* child *of a vertex* v *is a vertex of which* v *is the parent. A* leaf *is a vertex without children.*

4. *A rooted tree* $T = (V, E, r)$ *is called* finitely branching *if and only if any vertex has only a finite number of children.*

5. *A* branch *in a rooted tree* $T = (V, E, r)$ *is a finite or infinite sequence of vertices* $(a_0, a_1, \ldots)$ *such that* $a_0 = r$ *and* $(a_i, a_{i+1}) \in E$ *for any* $i + 1$ *which occurs as an index.*

Given a rooted tree $T = (V, E, r)$ we can pick any vertex $a \in V$ and consider the vertices above a in the sense of the partial order of T introduced above. The following lemma, whose proof we omit, states that this gives us a rooted tree with root a.

Lemma 89 *Assume that* $T = (V, E, r)$ *is a rooted tree and* $a \in V$*. If we define*

$$V_a := \{x \in V : a \sqsubseteq x\} \quad \text{and} \quad E_a := E \cap (V_a \times V_a),$$

then (V_a, E_a, a) *is a rooted tree with root* a*.*

König's lemma, which we are going to discuss now, is a famous combinatorial principle of mathematical logic. Its proof makes use of the pigeonhole principle and a certain form of choice. For us it will play a major role later in connection with the analysis of logical systems.

Theorem 90 (König's lemma) *Every infinite and finitely branching rooted tree has an infinite branch.*

PROOF Let $T = (V, E, r)$ be the given infinite and finitely branching rooted tree. By induction on n we pick for any natural number n a vertex a_n such that

- $(a_0, \ldots, a_n)$ is a branch in T,
- the set V_{a_n}, defined as in the lemma above, is infinite.

To begin with, we set $a_0 \coloneqq r$. Now assume that a_n has been defined. Then we know that $(a_0, \ldots, a_n)$ is a branch in T and that the set V_{a_n} is infinite. Since T is finitely branching, we also know that a_n has only finitely many children, say $b_1, \ldots, b_k$. As a consequence,

$$V_{a_n} = \{a_n\} \cup \bigcup_{i=1}^{k} V_{b_i}.$$

Hence Theorem 52 (the pigeonhole principle) implies that one of the sets $V_{b_1}, \ldots, V_{b_k}$ is infinite. Choose an $i \in \{1, \ldots, k\}$ for which V_{b_i} is infinite an set $a_{n+1} \coloneqq b_i$. Then, obviously, $(a_0, \ldots, a_{n+1})$ is a branch in T and $V_{a_{n+1}}$ is infinite.

Our way of selecting the vertices a_n then directly implies that $(a_0, a_1, \ldots)$ is the required infinite branch in T. □

The proof of this theorem is non-constructive. In choosing a_{n+1} we have to pick a child of a_n such that the subtree with root a_{n+1} is infinite. In general, we only know that at least one such subtree exists, but there is no constructive method to find out a suitable one.

Chapter 2

Classical propositional logic

What is logic and what is logic about? Let us begin with quoting *Wikipedia* for a first approach to logic.

Simple definitions of logic.

- The tool for distinguishing between the true and the false (Averroes).
- The science of reasoning, teaching the way of investigating unknown truth in connection with a thesis (Robert Kilwardby).
- The art whose function is to direct the reason lest it err in the manner of inferring or knowing (John Poinsot).
- The art of conducting reason well in knowing things (Antoine Arnauld).
- The right use of reason in the inquiry after truth (Isaac Watts).
- The science, as well as the art, of reasoning (Richard Whately).
- The science of the operations of the understanding which are subservient to the estimation of evidence (John Stuart Mill).
- The science of the laws of discursive thought (James McCosh).
- The science of the most general laws of truth (Gottlob Frege).
- The science which directs the operations of the mind in the attainment of truth (George Hayward Joyce).
- The branch of philosophy concerned with analysing the patterns of reasoning by which a conclusion is drawn from a set of premisses (Collins English Dictionary).

- The formal systematic study of the principles of valid inference and correct reasoning (Penguin Encyclopedia).

In the *Stanford Encyclopedia of Philosophy* the article on classical logic by Stewart Shapiro provides some further characteristics:

- Typically, a logic consists of a formal or informal language together with a deductive system and/or a model-theoretic semantics. The language is, or corresponds to, a part of a natural language like English or Greek. The deductive system is to capture, codify, or simply record which inferences are correct for the given language, and the semantics is to capture, codify, or record the meanings, or truth-conditions, or possible truth conditions, for at least part of the language.

- Today, logic is both a branch of mathematics and a branch of philosophy. In most large universities, both departments offer sequences of courses in logic, and there is usually a lot of overlap between them.

- Formal languages, deductive systems, and model-theoretic semantics are mathematical objects and, as such, the logician is interested in their mathematical properties and relations. Soundness, completeness, and most of the other results reported below are typical examples.

- Philosophically, logic is the study of correct reasoning. Reasoning is an epistemic, mental activity.

- This raises questions concerning the philosophical relevance of the mathematical aspects of logic. How do deducibility and validity, as properties of formal languages – sets of strings on a fixed alphabet – relate to correct reasoning? What do the mathematical results reported below have to do with the original philosophical issue? This is an instance of the philosophical problem of explaining how mathematics applies to non-mathematical reality.

Logic has a long history and its roots in philosophy. It was then taken up by mathematics, most prominently in algebraic form and in connection with questions concerning the foundations of mathematics. Later it began to play an important role in computer science, and a general slogan says that

logic is for computer science what analysis is for physics.

The following very brief and superficial overview mentions a few highlights in the development of logic:

- Aristoteles (384–322): development of some sort of theory of meaning, deduction and induction, syllogisms
- Wilhelm von Ockham (1285–1347): significant works on logic, physics, and theology (for example, Summa Logicae), Occam's razor
- Gottfried Wilhelm Leibniz (1646–1716): first steps in direction of algebraic logic, formalization of syllogistics within this framework
- George Boole (1815–1864): an algebraic calculus for logic, Boolean algebras, foundations of classical propositional logic
- Gottlob Frege (1848–1925): development of a formal language for formal proofs, foundations of classical predicate logic
- Bertrand Russell (1872–1970) und Alfred North Whitehead (1861–1947): Principia Mathematica (published in 1910), foundations of mathematics in a formal type-theoretic framework
- David Hilbert (1862–1943): the program of proof theory in answer to the foundational crisis of mathematics, formulas and proofs considered as bitstrings
- Alonzo Church (1903–1995): the lambda calculus as a formal abstraction of computations, undecidable problems
- Kurt Gödel (1906–1978): completeness and incompleteness results, provability logic
- Gerhard Gentzen (1909–1945): (co-)founder of modern proof theory, "proofs as programs"
- Alain Turing (1912–1954): Turing machines as universal models of computation, undecidability of the halting problem

Classical propositional logic is the most elementary logical system and provides the basis for many reasoning and argumentation systems. In this chapter we will discuss some of the principal properties of classical propositional logic. In doing this, we focus on a procedural point of view and emphasize algorithmic aspects of classical propositional logic.

Classical propositional logic in its present form goes back to George Boole who came up with a correspondence between logical symbols and algebraic operations and translated logic and logical questions into an algebraic formalism. By doing this, was able to apply the machinery of mathematics to solve certain logical problems.

Although syntax and semantics of classical propositional logic are very simple, classical propositional logic has considerable expressive power. In addition, it plays a crucial role in connection with the most central questions in algorithmic complexity theory.

2.1 The syntax of classical propositional logic CP

Definition 91 *The language $\mathcal{L}$ of classical propositional logic* CP *comprises the following syntactically different basic symbols:*

1. *Countably infinitely many atomic propositions $\mathfrak{p}_0, \mathfrak{p}_1, \mathfrak{p}_2, \ldots$ as well as the propositional constants $\top$ (true) and $\bot$ (false).*

2. *The logical connectives $\neg$ (negation), $\vee$ (disjunction), $\wedge$ (conjunction), and $\rightarrow$ (implication).*

As auxiliary symbols we have parentheses, brackets and commas. Atomic propositions correspond to basic assertions which will not be subdivided in our present context. Starting off from them and the propositional constants complex assertions are built up by means of the logical connectives.

Definition 92 *The formulas of $\mathcal{L}$ are inductively defined as follows:*

1. *Every atomic proposition and every propositional constant is a (so-called atomic) formula of $\mathcal{L}$.*

2. *If A and B are formulas of $\mathcal{L}$, then so are $(\neg A)$, $(A \vee B)$, $(A \wedge B)$, and $(A \rightarrow B)$ are also formulas of $\mathcal{L}$.*

The equivalence of two formulas A and B is expressed by stating that A implies B and vice versa, i.e.

$$(A \leftrightarrow B) \;:=\; ((A \rightarrow B) \wedge (B \rightarrow A)).$$

In addition, (finite) disjunctions and conjunctions of the formulas $A_1, \ldots, A_n$ are defined by induction on n as follows:

$$\bigvee_{i=1}^{0} A_i \;:=\; \bot \quad \text{and} \quad \bigvee_{i=1}^{n+1} A_i \;:=\; (\bigvee_{i=1}^{n} A_i \vee A_{n+1});$$
$$\bigwedge_{i=1}^{0} A_i \;:=\; \top \quad \text{and} \quad \bigwedge_{i=1}^{n+1} A_i \;:=\; (\bigwedge_{i=1}^{n} A_i \wedge A_{n+1});$$

In the following we will often simply speak of formulas if it is clear from the context that we refer to formulas of $\mathcal{L}$. In order to reduce the number of parentheses, we specify that (i) negation takes precedence over $\vee$, $\wedge$, and $\rightarrow$ and (ii) $\vee$ and $\wedge$ take

precedence over $\rightarrow$. Moreover, we often omit parentheses when there is no danger of confusion. According to this convention we may write, for example,

$$\mathfrak{p}_2 \vee \neg\mathfrak{p}_0 \rightarrow \mathfrak{p}_1$$

instead of

$$((\mathfrak{p}_2 \vee (\neg\mathfrak{p}_0)) \rightarrow \mathfrak{p}_1).$$

However, $\mathfrak{p}_0 \vee \mathfrak{p}_1 \wedge \mathfrak{p}_2$ does not stand for a formula since it is not clear whether it should abbreviate $((\mathfrak{p}_0 \vee \mathfrak{p}_1) \wedge \mathfrak{p}_2)$ or $(\mathfrak{p}_0 \vee (\mathfrak{p}_1 \wedge \mathfrak{p}_2))$.

Example 93 If the atomic propositions $\mathfrak{p}_0$ and $\mathfrak{p}_1$ stand for the statements "it rains" and "the street is wet", respectively, then the following compound statements have to be read as:

$\mathfrak{p}_0 \rightarrow \mathfrak{p}_1$	:	if it rains, the street is wet
$\mathfrak{p}_0 \wedge \mathfrak{p}_1$	:	it rains, and the street is wet
$\neg\mathfrak{p}_0 \rightarrow \bot$	:	if it does not rain, then falsum is the case
	:	(hence it rains)

We use certain categories of letters, possibly with subscripts or primed, as syntactic variables (metavariables) for certain syntactical categories. For the moment all we need is:

P, Q, R, S	for atomic formulas of $\mathcal{L}$,
A, B, C, D, E, F	for formulas of $\mathcal{L}$.

The length of a formula and the collection of its subformulas, which we introduce now, are important complexity measures for formulas which will be of some importance later.

Definition 94

1. *The* length $\ell(A)$ *of a formula A is the number of occurrences of atomic formulas and logical connectives in A.*

2. *The collection* $\mathrm{Sufo}(A)$ *of* subformulas *of a formula A is inductively defined as follows:*

   (a) *If A is an atomic formula, then* $\mathrm{Sufo}(A) := \{A\}$.

   (b) *If A is a formula $(\neg B)$, then* $\mathrm{Sufo}(A) := \{A\} \cup \mathrm{Sufo}(B)$.

   (c) *If A is a formula $(B \vee C)$, $(B \wedge C)$, or $(B \rightarrow C)$, then*

   $$\mathrm{Sufo}(A) := \{A\} \cup \mathrm{Sufo}(B) \cup \mathrm{Sufo}(C).$$

The element of $\mathrm{Sufo}(A)$ *are called the subformulas of* A*;* B *is a proper subformula of* A *in case it is a subformula of* A *different from* A*.*

The first assertion of the following lemma should be intuitively clear; its second assertion provides a useful bound for the number of subformulas of a given formula.

Lemma 95

1. *If* B *is a proper subformula of formula* A*, then the length of* B *is smaller than the length of* A*.*

2. *For all formulas* A *the number of elements of* $\mathrm{Sufo}(A)$ *is less than or equal to the length of* A*, i.e.*

$$|\mathrm{Sufo}(A)| \leq \ell(A).$$

The proof of this lemma is straightforward and left to the readers as an easy exercise.

2.2 The semantics of classical propositional logic CP

In this section we introduce the usual two-valued semantics for the classical propositional logic CP, based on the two truth values $\mathbf{t}$ (true) and $\mathbf{f}$ (false). We start off from valuations, which assign truth values $\mathbf{t}$ or $\mathbf{f}$ to all atomic propositions and then extend them to valuations of all formulas.

Definition 96 *A* valuation *is a function* $\mathfrak{V}$ *which assigns a truth value* $\mathfrak{V}(\mathfrak{p}_i) \in \{\mathbf{t}, \mathbf{f}\}$ *to any atomic proposition* $\mathfrak{p}_i$*, i.e.*

$$\mathfrak{V} : \{\mathfrak{p}_i : i \in \mathbb{N}\} \to \{\mathbf{f}, \mathbf{t}\}.$$

By following the inductive build-up of formulas, valuations of atomic propositions are easily extended to valuations of all formulas.

Definition 97 *Let* $\mathfrak{V}$ *be an arbitrary valuation. Then the truth value* $\widehat{\mathfrak{V}}(A) \in \{\mathbf{f}, \mathbf{t}\}$ *is inductively defined as follows:*

1. *For the atomic propositions we set* $\widehat{\mathfrak{V}}(\mathfrak{p}_i) := \mathfrak{V}(\mathfrak{p}_i)$ *for any* $i \in \mathbb{N}$*, for the propositional constants we set* $\widehat{\mathfrak{V}}(\bot) := \mathbf{f}$ *and* $\widehat{\mathfrak{V}}(\top) := \mathbf{t}$*.*

2. *If* A *is a formula* $(\neg B)$*, then*

$$\widehat{\mathfrak{V}}(A) := \begin{cases} \mathbf{f} & \text{if } \widehat{\mathfrak{V}}(B) = \mathbf{t}, \\ \mathbf{t} & \text{if } \widehat{\mathfrak{V}}(B) = \mathbf{f}. \end{cases}$$

3. If A is a formula $(B \vee C)$, then

$$\widehat{\mathfrak{V}}(A) := \begin{cases} \mathbf{f} & \text{if } \widehat{\mathfrak{V}}(B) = \mathbf{f} \text{ and } \widehat{\mathfrak{V}}(C) = \mathbf{f}, \\ \mathbf{t} & \text{if } \widehat{\mathfrak{V}}(B) = \mathbf{t} \text{ or } \widehat{\mathfrak{V}}(C) = \mathbf{t}. \end{cases}$$

4. If A is a formula $(B \wedge C)$, then

$$\widehat{\mathfrak{V}}(A) := \begin{cases} \mathbf{f} & \text{if } \widehat{\mathfrak{V}}(B) = \mathbf{f} \text{ or } \widehat{\mathfrak{V}}(C) = \mathbf{f}, \\ \mathbf{t} & \text{if } \widehat{\mathfrak{V}}(B) = \mathbf{t} \text{ and } \widehat{\mathfrak{V}}(C) = \mathbf{t}. \end{cases}$$

5. If A is a formula $(B \to C)$, then

$$\widehat{\mathfrak{V}}(A) := \begin{cases} \mathbf{f} & \text{if } \widehat{\mathfrak{V}}(B) = \mathbf{t} \text{ and } \widehat{\mathfrak{V}}(C) = \mathbf{f}, \\ \mathbf{t} & \text{if } \widehat{\mathfrak{V}}(B) = \mathbf{f} \text{ or } \widehat{\mathfrak{V}}(C) = \mathbf{t}. \end{cases}$$

Alternatively to this definition, we can also extend a valuation of the atomic propositions by computing the values of compound formulas with the aid of so-called truth tables.

Truth tables for classical propositional logic CP

A	$\neg A$
f	**t**
t	**f**

A	B	$A \vee B$	$A \wedge B$	$A \to B$
f	**f**	**f**	**f**	**t**
f	**t**	**t**	**f**	**t**
t	**f**	**t**	**f**	**f**
t	**t**	**t**	**t**	**t**

Example 98 Let $\mathfrak{V}$ and $\mathfrak{W}$ be two valuations such that

$$\begin{array}{lllllllll} \mathfrak{V}(\mathfrak{p}_0) & = & \mathbf{t}, & \mathfrak{V}(\mathfrak{p}_1) & = & \mathbf{f}, & \mathfrak{V}(\mathfrak{p}_2) & = & \mathbf{t}, \\ \mathfrak{W}(\mathfrak{p}_0) & = & \mathbf{f}, & \mathfrak{W}(\mathfrak{p}_1) & = & \mathbf{f}, & \mathfrak{W}(\mathfrak{p}_2) & = & \mathbf{t}. \end{array}$$

Then simple computations yield the following truth values:

$$\begin{array}{rcl@{\qquad}rcl} \widehat{\mathfrak{V}}(\mathfrak{p}_0 \vee \mathfrak{p}_1) & = & \mathbf{t}, & \widehat{\mathfrak{W}}(\mathfrak{p}_0 \vee \mathfrak{p}_1) & = & \mathbf{f}, \\ \widehat{\mathfrak{V}}(\neg(\mathfrak{p}_0 \wedge \mathfrak{p}_1)) & = & \mathbf{t}, & \widehat{\mathfrak{W}}(\neg(\mathfrak{p}_0 \wedge \mathfrak{p}_1)) & = & \mathbf{t}, \\ \widehat{\mathfrak{V}}(\neg(\mathfrak{p}_2 \wedge \neg(\neg\mathfrak{p}_0 \vee \mathfrak{p}_1))) & = & \mathbf{f}, & \widehat{\mathfrak{W}}(\neg(\mathfrak{p}_2 \wedge \neg(\neg\mathfrak{p}_0 \vee \mathfrak{p}_1))) & = & \mathbf{t}. \end{array}$$

What one also should observe from this example is that that the truth value of a compound formula depends only on the truth values of the atomic propositions occurring in this formula.

Definition 99

1. *A formula A is called* valid *if and only if we have $\widehat{\mathfrak{V}}(A) = \mathbf{t}$ for all valuations $\mathfrak{V}$; in this case we write $\vDash A$.*

2. *A formula A is called* satisfiable *if and only if there exists a valuation $\mathfrak{V}$ such that $\widehat{\mathfrak{V}}(A) = \mathbf{t}$; a formula A is called* unsatisfiable *if A is not satisfiable.*

3. *Formulas A and B are called* equivalent *if we have $\widehat{\mathfrak{V}}(A) = \widehat{\mathfrak{V}}(B)$ for all valuations $\mathfrak{V}$.*

In the next section we will address the problem of deciding whether a given formula is satisfiable. As the following lemma, whose proof is again left to the reader as an easy exercise, tells us that testing for satisfiability is closely related to testing for validity and equivalence.

Lemma 100

1. *A formula A is valid if and only if $\neg A$ is unsatisfiable.*

2. *A formula A is valid if and only if it is equivalent to $\top$.*

3. *A formula A is unsatisfiable if and only if it is equivalent to $\bot$.*

4. *Formulas A and B are equivalent if and only if $A \leftrightarrow B$ is valid.*

We end this section with a theorem which states some important equivalences. In particular, it shows that in principle we could have done with less logical connectives. For example, $\neg$ and $\vee$ are sufficient to obtain all formulas modulo equivalence.

Theorem 101

1. *Formula $A \to B$ is equivalent to formula $\neg A \vee B$; formula $A \wedge B$ is equivalent to formula $\neg(\neg A \vee \neg B)$.*
2. *Formula $A \vee \neg A$ is equivalent to formula $\top$; formula $A \wedge \neg A$ is equivalent to formula $\bot$.*
3. *For any formula A there exists an equivalent formula B with the same atomic propositions in which the logical connectives $\wedge$ and $\to$ do not occur.*

PROOF For proving the first part of this theorem, we simply consider the following truth tables:

A	B	$A \to B$	$\neg A \vee B$	$\neg A \vee \neg B$	$\neg(\neg A \vee \neg B)$	$A \wedge B$
f	**f**	**t**	**t**	**t**	**f**	**f**
f	**t**	**t**	**t**	**t**	**f**	**f**
t	**f**	**f**	**f**	**t**	**f**	**f**
t	**t**	**t**	**t**	**f**	**t**	**t**

The second part of this theorem is obvious. Its third part follows from the first if we successively replace all subformulas of A of the form $C \to D$ by $\neg C \vee D$ and all those of the form $C \wedge D$ by $\neg(\neg C \vee \neg D)$. □

2.3 Testing for satisfiability

In this section we introduce a first method for testing propositional formulas for satisfiability and validity. We also make some observations concerning the computational complexity of this method with the result that it is exponential in the number of atomic proposition which occur in the formula to be tested.

Definition 102 *Given a formula A, we write* $\mathrm{Atp}(A)$ *for the collection of all atomic propositions which occur in A.*

The proof of the following lemma is trivial and proved by a straightforward induction on the build-up of the formula in question; we omit the details.

Lemma 103 *Given an arbitrary formula A as well as two valuations $\mathfrak{V}$ and $\mathfrak{W}$ such that $\mathfrak{V}(P) = \mathfrak{W}(P)$ for all $P \in \mathrm{Atp}(A)$, we have $\widehat{\mathfrak{V}}(A) = \widehat{\mathfrak{W}}(A)$.*

Although trivial, this lemma tells us that the evaluation of a formula A only depends on the finitely many ways, the truth values can be distributed on the atomic propositions occurring in A.

Definition 104 *Given atomic propositions $P_1, \ldots, P_n$ and truth values $a_1, \ldots, a_n$ we introduce a valuation $\mathfrak{X}_{(P_1|a_1,\ldots,P_n|a_n)}$ by setting*

$$\mathfrak{X}_{(P_1|a_1,\ldots,P_n|a_n)}(Q) := \begin{cases} a_i & \text{if } Q \text{ is the atomic proposition } P_i, \\ \mathbf{f} & \text{if } Q \notin \{P_1, \ldots, P_n\}. \end{cases}$$

This is a compact form of notation useful for formulating the algorithm `?-sat(.)` below. It simply tells us that the given atomic propositions $P_1, \ldots, P_n$ have to be given the truth values $a_1, \ldots, a_n$, respectively.

Example 105 If A is the formula $P \to Q \wedge R$, then we have, for example,

$$\widehat{\mathfrak{X}}_{(P|\mathbf{t},Q|\mathbf{f},R|\mathbf{f})}(A) \;=\; \widehat{\mathfrak{X}}_{(P|\mathbf{t},Q|\mathbf{f})}(A) \;=\; \mathbf{f}.$$

Now we turn to testing a formula A for satisfiability. The conceptually most elementary method proceeds as follows: If $\mathrm{Atp}(A) = \{P_1, \ldots, P_n\}$, we consider all possible distributions of truth values to the atomic propositions $P_1, \ldots, P_n$. If for at least one such distribution the value of A computes to $\mathbf{t}$, then A is satisfiable; otherwise it is unsatisfiable.

Test for satisfiability `?-sat(.)`

Input: Formula A

Step 1: Determine all atomic propositions which occur in A and write them as a list without repetitions: $P_1, \ldots, P_n$.

Step 2: `FOR ALL` $a_1, \ldots, a_n$ `DO`
determine $\widehat{\mathfrak{X}}_{(P_1|a_1,\ldots,P_n|a_n)}(A)$;
`IF` $\widehat{\mathfrak{X}}_{(P_1|a_1,\ldots,P_n|a_n)}(A) =$ `t THEN RETURN ‘‘satisfiable’’.`

Step 3: `RETURN ‘‘not satisfiable’’.`

Although conceptually very simple, the algorithm `?-sat(.)` is very problematic from the point of view of computational complexity. Suppose that the formula A contains n atomic propositions $P_1, \ldots, P_n$ and suppose, in addition, that A is unsatisfiable. Then `?-sat(.)` has to test all possible truth assignments to $P_1, \ldots, P_n$, i.e. 2^n many truth assignments. As a consequence, the worst case runtime of `?-sat(.)` applied to a formula A can be exponential in the number of elements of $\mathrm{Atp}(A)$.

Example 106 Consider the formula $A := P \wedge (Q \wedge (\neg R \vee \neg P))$. Then $\mathrm{Atp}(A)$ is the collection $\{P, Q, R\}$, and `?-sat(.)` gives us:

$$\begin{array}{llll} 1: & \widehat{\mathfrak{X}}_{(P|\mathbf{f},Q|\mathbf{f},R|\mathbf{f})}(A) = \mathbf{f}; & 5: & \widehat{\mathfrak{X}}_{(P|\mathbf{t},Q|\mathbf{f},R|\mathbf{f})}(A) = \mathbf{f}; \\ 2: & \widehat{\mathfrak{X}}_{(P|\mathbf{f},Q|\mathbf{f},R|\mathbf{t})}(A) = \mathbf{f}; & 6: & \widehat{\mathfrak{X}}_{(P|\mathbf{t},Q|\mathbf{f},R|\mathbf{t})}(A) = \mathbf{f}; \\ 3: & \widehat{\mathfrak{X}}_{(P|\mathbf{f},Q|\mathbf{t},R|\mathbf{f})}(A) = \mathbf{f}; & 7: & \widehat{\mathfrak{X}}_{(P|\mathbf{t},Q|\mathbf{t},R|\mathbf{f})}(A) = \mathbf{t}. \\ 4: & \widehat{\mathfrak{X}}_{(P|\mathbf{f},Q|\mathbf{t},R|\mathbf{t})}(A) = \mathbf{f}; & & \end{array}$$

Remark 107 The algorithm `?-sat(.)` can also be used in order to check whether a formula A is valid. In view of Lemma 100 we simply proceed as follows:

(1) Apply `?-sat(.)` to $\neg A$.

(2) If `?-sat(.)` gives the answer "not satisfiable", then A is valid.

(3) If `?-sat(.)` gives the answer "satisfiable", then A is not valid.

Now we present formulas PH_n with parameter n which are valid, but whose validity is difficult to verify in propositional logic by most known methods. In particular, when trying to validate these formulas by the methods introduced so far we see an "explosion" of the computational complexity if n increases.

Example 108 (Pigeonhole/n)

Statement: If $n+1$ items are put into n pigeonholes, then at least one pigeonhole must contain more than one item.

By Theorem 52 we know that this assertion is correct, and the proof we gave, was more or less trivial by induction on n. What we do now, is to formulate this assertion in classical propositional logic.

Formalization in CP: We choose atomic propositions

$$P_{i,j} \qquad \text{for} \quad 1 \le i \le n+1 \quad \text{and} \quad 1 \le j \le n$$

which we interpret as

$$P_{i,j} \quad :\Leftrightarrow \quad \text{item } i \text{ goes into hole } j.$$

Then we introduce the following abbreviations:

(i) "Every item is in one hole."

$$\mathrm{IH}_n := \bigwedge_{i=1}^{n+1} (P_{i,1} \vee \ldots \vee P_{i,n})$$

(ii) "Hole j contains at least two items."

$$\mathrm{TI}_n^j \coloneqq \bigvee_{1 \le i < k \le n+1} (P_{i,j} \wedge P_{k,j})$$

(iii) "At least one hole contains at least two items."

$$\mathrm{TI}_n \coloneqq \bigvee_{j=1}^{n} \mathrm{TI}_n^j$$

(iv) Pigeonhole/n

$$\mathrm{PH}_n \coloneqq \mathrm{IH}_n \to \mathrm{TI}_n$$

As mentioned above, the assertion PH_n is correct for any natural number $n \geq 1$ according to Theorem 52. It was proved then by simple complete induction on n. As a consequence, we know that the propositional formula $\neg\mathrm{PH}_n$ is unsatisfiable.

But now observe that in checking, for example, $\neg\mathrm{PH}_4$ by our algorithm `?-sat(.)`, $2^{20} = 1'048'576$ different valuations of the 20 atomic propositions occurring in PH_4 have to be tested. Imagine what this means for 10 holes and 11 items!

2.4 Theories

Often we are confronted with the situation that we are given a collection of data and want to know whether a specific assertion follows from these data. If the data and the assertion, we are interested in, can be represented in the language of classical propositional logic, we are in a position to use our formalism to treat this problem in a proper way.

The basic idea is very simple: the given data provide a so-called theory T, and then we check whether the assertion A we have to verify is a logical consequence of T in the sense below.

Definition 109

1. Theories *are finite or infinite collections of formulas.*
2. *A valuation* $\mathfrak{V}$ satisfies *a theory* T *if we have* $\widehat{\mathfrak{V}}(A) = \mathtt{t}$ *for all* $A \in T$*; in this case we write* $\mathfrak{V} \vDash T$*.*
3. *A theory* T *is called* satisfiable *if there exists a valuation which satisfies* T*; a theory* T *is called* unsatisfiable *if* T *is not satisfiable.*

4. *A formula A is a* logical consequence *of T if we have $\widehat{\mathfrak{V}}(A) = \mathbf{t}$ for all valuations $\mathfrak{V}$ which satisfy T; in this case we write $T \models A$.*

Remark 110 From this definition and Definition 99 we immediately obtain that for the empty theory T and all formulas A we have

$$T \models A \iff \models A.$$

It is also clear from the previous definition that the notion of logical consequence is monoton in the sense that

$$T_1 \subseteq T_2 \text{ and } T_1 \models A \implies T_2 \models A$$

for all theories T_1 and T_2 and all formulas A. The following lemma presents some further properties of logical validity.

Lemma 111 *For all theories T and formulas A, B we have:*

1. $T \models A \iff T \cup \{\neg A\}$ *is unsatisfiable.*
2. $T \models A$ *and* $T \models A \to B \implies T \models B$.

The proof of this lemma is left to the reader as an exercise. A further property of logical validity is that it satisfies the so-called deduction theorem which allows to reduce logical consequences of a theory to logical validity.

Theorem 112 (Deduction theorem)
For all theories T and all formulas A, B we have:

1. $T \cup \{A\} \models B \iff T \models A \to B$.
2. $T \models A$ *and* $T \cup \{A\} \models B \implies T \models B$.
3. *If T is the theory $\{A_1, \ldots, A_n\}$, then*

$$T \models B \iff \models (A_1 \wedge \ldots \wedge A_n) \to B.$$

Proof To show the direction from left to right of the first assertion, we assume $T \cup \{A\} \models B$ and take a valuation $\mathfrak{V}$ such that $\mathfrak{V} \models T$. Now we distinguish two cases:

(i) $\widehat{\mathfrak{V}}(A) = \mathbf{f}$. Then $\widehat{\mathfrak{V}}(A \to B) = \mathbf{t}$.

(ii) $\widehat{\mathfrak{V}}(A) = \mathbf{t}$. Because of $\mathfrak{V} \models T$ we then even have $\mathfrak{V} \models T \cup \{A\}$. As a consequence, $T \cup \{A\} \models B$ implies $\widehat{\mathfrak{V}}(B) = \mathbf{t}$, and therefore $\widehat{\mathfrak{V}}(A \to B) = \mathbf{t}$.

Hence we have seen that $\widehat{\mathfrak{V}}(A \to B) = \mathbf{t}$ for any valuation $\mathfrak{V}$ with $\mathfrak{V} \models T$.

The converse direction is established similarly. Assume $T \models A \to B$ and let $\mathfrak{V}$ be a valuation such that $\mathfrak{V} \models T \cup \{A\}$. Then we also have $\mathfrak{V} \models T$ and $\widehat{\mathfrak{V}}(A) = \mathbf{t}$. Thus $T \models A \to B$ implies $\widehat{\mathfrak{V}}(A \to B) = \mathbf{t}$; which means that we also have $\widehat{\mathfrak{V}}(B) = \mathbf{t}$. Altogether we have seen that $\widehat{\mathfrak{V}}(B) = \mathbf{t}$ for any valuation $\mathfrak{V}$ with $\mathfrak{V} \models T \cup \{A\}$.

The second assertion of this lemma is an immediate consequence of the first assertion and the previous lemma. The third assertion follows directly from the first. □

Example 113 (Light on/off)

General situation: An automated office protection system is activated if and only if there is money in the office or nobody is in the office. Furthermore, the light has to be switched on, if it is not the case that there is nobody in the office and the automated office protection system is not activated.

Question: An employee now decides to leave the light switched on. Is this behavior correct?

Formalization: We select four atomic propositions P, Q, R, S which we interpret them as follows:

P	:::	the office protection system is activated,
Q	:::	there is money in the office,
R	:::	nobody is in the office,
S	:::	the light is switched on.

The general situation is then represented by the following theory

$$\mathrm{SIT} := \{\, P \to (Q \vee R),\ (Q \vee R) \to P,\ \neg(R \wedge \neg P) \to S \,\}.$$

Hence we want to find out whether

$$\mathrm{SIT} \models S.$$

In view of Lemma 111.1 this is equivalent to

$$\mathrm{SIT} \cup \{\neg S\} \text{ is unsatisfiable.}$$

Assumption: $\mathrm{SIT} \cup \{\neg S\}$ is satisfiable.

Then there exists a valuation $\mathfrak{V}$ such that

(1) $$\widehat{\mathfrak{V}}(\neg S) = \mathbf{t},$$

(2) $$\widehat{\mathfrak{V}}(P \to (Q \vee R)) = \mathbf{t},$$

(3) $$\widehat{\mathfrak{V}}((Q \vee R) \to P) = \mathbf{t},$$

(4) $$\widehat{\mathfrak{V}}(\neg(R \wedge \neg P) \to S) = \mathbf{t}.$$

From (1) and (4) we obtain

(5) $$\widehat{\mathfrak{V}}(\neg(R \wedge \neg P)) = \mathbf{f}, \quad \text{i.e.} \quad \widehat{\mathfrak{V}}(R \wedge \neg P) = \mathbf{t},$$

and therefore

(6) $$\widehat{\mathfrak{V}}(R) = \mathbf{t},$$

(7) $$\widehat{\mathfrak{V}}(P) = \mathbf{f}.$$

Now (3) and (7) imply

(8) $$\widehat{\mathfrak{V}}(Q \vee R) = \mathbf{f}.$$

However, (8) yields $\widehat{\mathfrak{V}}(R) = \mathbf{f}$, contradicting (6). This shows that our assumption is wrong.

Hence S is a logical consequence of SIT, and the employee's behavior is correct.

2.5 A Hilbert calculus HC for CP

Now we turn to some deductive aspects of CP. Our eventual aim is to characterize the logical consequences of a given theory T in a procedural way and to develop an efficient procedure for checking whether an assertion A follows from T.

This so-called axiomatic method has been propagated by David Hilbert (1862–1943), and so some of these calculi are called Hilbert calculi. Since they provide a conceptually very simple framework for formal derivations, we begin with presenting a Hilbert calculus HC for CP. However, from the point of view of proof search, Hilbert calculi have grave disadvantages. Therefore we will introduce an alternative to HC later.

Definition 114 *A* Hilbert calculus $\mathcal{H}$ *consists of a collection* $\mathrm{Ax}(\mathcal{H})$ *of formulas and a collection* $\mathrm{Ru}(\mathcal{H})$ *of configurations of the form*

$$\frac{A_1 \quad A_2 \quad \ldots \quad A_n}{B}.$$

The elements of $\mathrm{Ax}(\mathcal{H})$ *are called the* axioms *of* $\mathcal{H}$*, the elements of* $\mathrm{Ru}(\mathcal{H})$ *are called its* rules of inference. *Given a rule of inference*

$$\frac{A_1 \quad A_2 \quad \ldots \quad A_n}{B},$$

then the formulas $A_1, \ldots, A_n$ *are the* premises *and the formula* B *is the conclusion of this rule.*

Given a Hilbert calculus $\mathcal{H}$ there is an easy and perspicuous way to define what it means that a formula is derivable in the sense of $\mathcal{H}$ from a theory T.

Definition 115 *Let* $\mathcal{H}$ *be a Hilbert calculus and let* T *be a theory.*

1. *A* T-proof *in* $\mathcal{H}$ *is a finite sequence of formulas*

   $$A_0, \ldots, A_n$$

   which has the following property: Every member A_i *of this sequence is an element of* $\mathrm{Ax}(\mathcal{H})$*, an element of* T*, or the conclusion of a rule from* $\mathrm{Ru}(\mathcal{H})$ *such that all premises of this rule occur left of* A_i *in this sequence.*

2. *We say that formula* A *is* derivable *in* $\mathcal{H}$ *from* T *if there exists a* T*-proof in* $\mathcal{H}$ *whose last member is the formula* A*. Then we write* $T \vdash_{\mathcal{H}} A$*.*

3. *If* T *is the empty set, we speak of proofs in* $\mathcal{H}$ *and derivability in* $\mathcal{H}$*. Accordingly, we write* $\vdash_{\mathcal{H}} A$ *if* A *is derivable in* $\mathcal{H}$ *from the empty set.*

Often we also use formulations like "A is provable in $\mathcal{H}$" as synonyms of "A is derivable in $\mathcal{H}$".

In the following we present one specific Hilbert calculus HC for CP. It is a system which comprises many axioms, but only one type of rules of inference, the so-called *Modus Ponens*. HC is characterized by the following axioms and rules of inference.

Axioms of HC**.** For any formulas A, B, C:

(A1) $\quad \bot \to A,$

(A2) $\quad A \to \top,$

(A3) $A \to A$,

(A4) $(A \to B) \to ((B \to C) \to (A \to C))$,

(A5) $(A \to (B \to C)) \to ((A \wedge B) \to C)$,

(A6) $((A \wedge B) \to C) \to (A \to (B \to C))$,

(A7) $A \to (A \vee B)$,

(A8) $B \to (A \vee B)$,

(A9) $(A \to C) \to ((B \to C) \to ((A \vee B) \to C))$,

(A10) $(A \wedge B) \to A$,

(A11) $(A \wedge B) \to B$,

(A12) $(A \to B) \to ((A \to C) \to (A \to (B \wedge C)))$,

(A13) $(A \to B) \to (\neg B \to \neg A)$,

(A14) $A \to (\neg A \to B)$,

(A15) $(A \to (A \wedge \neg A)) \to \neg A$,

(A16) $A \vee \neg A$.

Rules of inference of HC. For any formulas A, B:

(MP) $$\frac{A \qquad A \to B}{B}$$

The next aim is to show that, given a theory T, all formulas which are derivable in HC from T are logical consequences of T. To achieve this, we first convince ourselves that all axioms of HC are valid.

Lemma 116 *Any axiom of* HC *is valid.*

PROOF It is a simple exercise to show, for example, by means of the truth tables that $\widehat{\mathfrak{V}}(A) = \mathbf{t}$ for any axiom of HC and any valuation $\mathfrak{V}$. Details are left to the reader. □

This lemma together with Lemma 111.2, which states that the notion of logical consequence is closed under the Modus Ponens, immediately gives us the correctness theorem for HC.

Theorem 117 (Correctness of HC)

Let T be a theory. Then any formula A which is derivable in HC *from T is a logical consequence of T; i.e.*

$$T \vdash_{\mathsf{HC}} A \quad \Longrightarrow \quad T \vDash A.$$

PROOF Given a formula A which is derivable in HC from theory T, we know that there exists a T-proof

$$A_0, \ldots, A_n$$

in HC whose last formula A_n is our formula A. Making use of the previous lemma and Lemma 111.2 it is now a straightforward affair to show by complete induction on i that $T \vDash A_i$ for any i, $0 \leq i \leq n$. This implies our assertion. □

The converse of this theorem – called the completeness of HC – is correct as well: if A is a logical consequence of a theory T, then A is derivable in HC from T. However, we dispense with a proof of this theorem now and deduce it from the completeness result for the proof search calculus PSC which we will introduce later.

Theorem 118 (Completeness of HC)

Let T be a theory. Then any formula A which is a logical consequence of T is derivable in HC *from T; i.e.*

$$T \vDash A \quad \Longrightarrow \quad T \vdash_{\mathsf{HC}} A.$$

From the correctness and completeness of HC we obtain the crucial result that the Hilbert calculus HC is an adequate formalism to syntactically characterize the logical consequences of a theory.

Corollary 119 *For all theories T and formulas A we have*

$$T \vdash_{\mathsf{HC}} A \quad \Longleftrightarrow \quad T \vDash A.$$

A nice consequence of this result is the finiteness property of logical consequence: given a theory T and a logical consequence A of T, already a finite part of T is sufficient to imply A.

Corollary 120 (Finiteness property)

Let T be an arbitrary theory. If A is a logical consequence of T, then there exist finitely many $B_1, \ldots, B_n \in T$ such that

$$\vDash (B_1 \wedge \ldots \wedge B_n) \to A.$$

PROOF If A is a logical consequence of T, then by Theorem 118 A is derivable in HC from T. Pick a T-proof in HC whose last formula is the formula A and let $B_1, \ldots, B_n$ be those elements of T which occur in this proof. □

2.6 Negation normal forms

In order to simplify the structure of formulas and to reduce the number of connectives we have to deal with in proof search (see next section) we now introduce the notion of negation normal form and convince ourselves that any formula can be transformed into negation normal form in reasonable time.

A formula is said to be in negation normal form if negations occurs only immediately in front of atomic propositions and if $\neg$, $\vee$, and $\wedge$ are the only permitted logical connectives. Formulas are transformed into negation normal form by first eliminating all implications, then pushing negations inside by using De Morgan's laws, and finally eliminating all double negations.

Definition 121 *A formula A is in* negation normal form *if the connective $\rightarrow$ does not occur in A and if for all subformulas $\neg B$ of A the formula B is an atomic proposition.*

Example 122

1. The following formulas are in negation normal form:

$$\top, \quad \bot, \quad P, \quad \neg P, \quad \neg P \vee Q, \quad \neg P \vee (\neg Q \wedge P).$$

2. The following formulas are not in negation normal form:

$$\neg\top, \quad \neg\bot, \quad \neg\neg P, \quad P \rightarrow Q, \quad \neg P \vee \neg(P \wedge Q).$$

Now we turn to the transformation of a formula A into an equivalent formula in negation normal form. As mentioned above, the first step is to eliminate all implications. This is easily achieved by making use of Theorem 101.1.

Definition 123 *The $\rightarrow$-reduction $\rho(A)$ of a formula A is inductively defined as follows:*

1. *If A is atomic, then $\rho(A) := A$.*
2. *If A is a formula $\neg B$, then $\rho(A) := \neg\rho(B)$.*
3. *If A is a formula $B \vee C$, then $\rho(A) := \rho(B) \vee \rho(C)$.*
4. *If A is a formula $B \wedge C$, then $\rho(A) := \rho(B) \wedge \rho(C)$.*
5. *If A is a formula $B \rightarrow C$, then $\rho(A) := \neg\rho(B) \vee \rho(C)$.*

In view of Theorem 101.1 the following lemma is proved by straightforward induction on the build-up of the formula A.

Lemma 124 *For any formula A, its $\rightarrow$-reduction $\rho(A)$ is equivalent to A and does not contain implications. In addition,* $\mathrm{Atp}(\rho(A)) = \mathrm{Atp}(A)$.

The next step is to move negations inside. This is achieved by employing De Morgan's laws and the law of double negation, namely

$$\begin{aligned} \neg(A \vee B) &\quad \text{equivalent to} \quad (\neg A \wedge \neg B), \\ \neg(A \wedge B) &\quad \text{equivalent to} \quad (\neg A \vee \neg B), \\ \neg\neg A &\quad \text{equivalent to} \quad A. \end{aligned}$$

These equivalences are easily checked by truth table calculations.

Definition 125 *The NNF-reduction $\nu(A)$ of a formula A without implications is inductively defined as follows:*

1. *If A is an atomic proposition, the negation of an atomic proposition, or a propositional constant, then $\nu(A) := A$.*
2. *If A is the formula $\neg\top$, then $\nu(A) := \bot$; if it is the formula $\neg\bot$, then $\nu(A) := \top$.*
3. *If A is a formula $\neg\neg B$, then $\nu(A) := \nu(B)$.*
4. *If A is a formula $B \vee C$, then $\nu(A) := \nu(B) \vee \nu(C)$.*
5. *If A is a formula $\neg(B \vee C)$, then $\nu(A) := \nu(\neg B) \wedge \nu(\neg C)$.*
6. *If A is a formula $B \wedge C$, then $\nu(A) := \nu(B) \wedge \nu(C)$.*
7. *If A is a formula $\neg(B \wedge C)$, then $\nu(A) := \nu(\neg B) \vee \nu(\neg C)$.*

As before, a straightforward induction on the build-up on A yields the following reduction lemma.

Lemma 126 *For any formula A without implications, its NNF-reduction $\nu(A)$ is equivalent to A and in negation normal form. In addition,* $\mathrm{Atp}(\nu(A)) = \mathrm{Atp}(A)$.

Definition 127 *Given a formula A, its* negation normal form $\mathrm{nnf}(A)$ *is defined by*

$$\mathrm{nnf}(A) := \nu(\rho(A)).$$

It is easy to see that for any formula A which is in negation normal form, the formulas $\mathrm{nnf}(A)$ and A are identical.

The negation of a formula A, which itself is in negation normal form, has not to be in negation normal form. Consider, for example, $P \vee Q$ and $\neg(P \vee Q)$. However, it is possible to assign to any A in negation normal form a formula $\overline{A}$ which is in negation normal form and equivalent to $\neg A$.

Definition 128 *The* complement $\overline{A}$ *of a formula A in negation normal form is inductively defined as follows:*

1. *For all atomic propositions and their negations we set*

$$\overline{P} := \neg P, \quad \overline{\neg P} := P.$$

2. *For propositional constants and their negations we set:*

$$\overline{\top} := \bot, \quad \overline{\bot} := \top.$$

3. *If A is a formula $B \vee C$, then $\overline{A} := \overline{B} \wedge \overline{C}$; if A is a formula $B \wedge C$, then $\overline{A} := \overline{B} \vee \overline{C}$.*

Again, straightforward induction on the build-up of A gives us the following lemma about the complements of formulas in negation normal form.

Lemma 129 *Let A be a formula in negation normal form. Then the complement $\overline{A}$ of A is in negation normal form as well and equivalent to $\neg A$, and we have $\mathrm{Atp}(A) = \mathrm{Atp}(\overline{A})$. In addition, $\overline{\overline{A}}$ is identical to A.*

The transformation of a formula A into the equivalent formula $\mathrm{nnf}(A)$ and its complement $\overline{\mathrm{nnf}(A)}$ can be carried through with time-complexity polynomial in $\ell(A)$. From a complexity-theoretic point of view it is therefore justified when dealing with formulas to first transform them into negation normal form and then to work in this restricted environment.

2.7 The proof search calculus PSC

Legitimated by what we have seen in the previous section, we confine ourselves to carry through proof search for formulas in negation normal form. If we want to find out whether a formula A in negation normal form is valid, we have to distinguish two different situations:

- A conjunction $A \wedge B$ is valid if and only if A and B are valid. Hence the task of testing whether $A \wedge B$ is valid can be reduced to the two "smaller" tasks (with respect to the length of the formulas) of testing whether A and B are valid.

- On the other hand, a disjunction $A \vee B$ is valid if A is valid or B is valid. However, it may also happen that $A \vee B$ is valid and none of A and B is valid. Take, for example, the formula $P \vee \neg P$. Hence in the case of disjunctions $A \vee B$ it in general is not possible to reduce the task of testing whether $A \vee B$ is valid to the "smaller" tasks of testing whether A or B is valid.

To overcome this problem and to provide a better framework for systematic proof search, our calculus PSC does not derive individual formulas but finite sequences of formulas.

In the following we use the capital Greek letters Γ, Δ, Π, Σ, possibly with subscripts or primes, as syntactic variables (metavariables) for finite (possibly empty) sequences of formulas in negation normal form. Further notation:

- If Γ is the sequence $A_1, \ldots, A_n$, then we write $\mathrm{set}(\Gamma)$ for the corresponding set of formulas $\{A_1, \ldots, A_n\}$ and $\Gamma^\vee$ for the disjunction formed by these formulas, i.e.

$$\Gamma^\vee \;:=\; A_1 \vee \ldots \vee A_n \;:=\; \bigvee_{i=1}^{n} A_i.$$

  Hence, if Γ is the empty sequence, then $\mathrm{set}(\Gamma) = \varnothing$ and $\Gamma^\vee$ is the formula $\bot$.

- If Γ is the sequence $A_1, \ldots, A_m$ and Δ the sequence $B_1, \ldots, B_n$, then we write Γ, Δ for the sequence $A_1, \ldots, A_m, B_1, \ldots, B_n$.

- We can also combine sequences and formulas: If Γ is the sequence $A_1, \ldots, A_m$ and Δ the sequence $B_1, \ldots, B_n$, then we write Γ, C, Δ for the sequence

$$A_1, \ldots, A_m, C, B_1, \ldots, B_n.$$

  More complicated expressions like, for example, $\Gamma, C_1, C_2, \Delta, D\Pi$ are to be understood accordingly.

We interpret finite sequences Γ disjunctively, meaning that semantically the sequence Γ is identified with the formula $\Gamma^\vee$. Given a valuation $\mathfrak{V}$, we therefore set

$$\widehat{\mathfrak{V}}(\Gamma) \;:=\; \widehat{\mathfrak{V}}(\Gamma^\vee);$$

accordingly, the sequence Γ is called valid – denoted by $\vDash \Gamma$ – if $\widehat{\mathfrak{V}}(\Gamma) = \mathbf{t}$ for all valuations $\mathfrak{V}$.

Since PSC derives finite sequences of formulas rather than individual formulas, we have to generalize our notions of axiom and rules of inference accordingly.

Axioms of PSC. For all finite sequences Γ, Δ, Π of formulas in negation normal form and all formulas A which are an atomic proposition or the complement of an atomic proposition:

(True) $$\Gamma, \top, \Delta$$

(Id) $$\Gamma, A, \Delta, \overline{A}, \Pi$$

Rules of inference of PSC. For all finite sequences Γ, Δ of formulas in negation normal form and all formulas A, B in negation normal form:

$(\vee)$ $$\frac{\Gamma, A, B, \Delta}{\Gamma, A \vee B, \Delta}$$

$(\wedge)$ $$\frac{\Gamma, A, \Delta \qquad \Gamma, B, \Delta}{\Gamma, A \wedge B, \Delta}$$

The sequence Γ, A, B, Δ is the premise of the rule $(\vee)$ with conclusion $\Gamma, A \vee B, \Delta$; the sequences Γ, A, Δ and Γ, B, Δ are the premises of the rule $(\wedge)$ with conclusion $\Gamma, A \wedge B, \Delta$.

Based on these notions of axiom and rule of inference we can now proceed as in the case of a Hilbert calculus and introduce derivability in PSC.

Definition 130

1. *A* proof *in* PSC *is a finite sequence of sequences*

$$\Gamma_0, \ldots, \Gamma_n$$

of formulas in negation normal form which has the following property: Every member Γ_i of this sequence is an axiom of PSC *or the conclusion of a rule $(\vee)$ or $(\wedge)$ such that all premises of this rule occur left of Γ_i in this sequence.*

2. *We say that Γ is* derivable *in* PSC *if there exists a proof in* PSC *whose last member is the sequence Γ. Then we write $\vdash_{\mathsf{PSC}} \Gamma$.*

Often we also use formulations like "Γ is provable in PSC" as synonyms of "Γ is derivable in PSC".

It is an obvious observation that all axioms of PSC are valid in the sense described above and that the rules of inference of PSC preserve validity. The proof of the following lemma is very easy and can be omitted.

Lemma 131

1. *If* Γ *is an axiom of* PSC *and* $\mathfrak{V}$ *a valuation, then* $\widehat{\mathfrak{V}}(\Gamma) = \mathbf{t}$.

2. *For all sequences* Γ, Δ *of formulas in negation normal, all formulas* A, B *in negation normal form, and all valuations* $\mathfrak{V}$ *we have:*

   (a) $\widehat{\mathfrak{V}}(\Gamma, A, B, \Delta) = \mathbf{t} \implies \widehat{\mathfrak{V}}(\Gamma, A \vee B, \Delta) = \mathbf{t}$.

   (b) $\widehat{\mathfrak{V}}(\Gamma, A, \Delta) = \mathbf{t}$ *and* $\widehat{\mathfrak{V}}(\Gamma, B, \Delta) = \mathbf{t} \implies \widehat{\mathfrak{V}}(\Gamma, A \wedge B, \Delta) = \mathbf{t}$.

As in the case of the Hilbert calculus HC, this lemma directly implies correctness: If we have a valuation $\mathfrak{V}$ and a proof $\Gamma_0, \ldots, \Gamma_n$ in PSC, then simply use the previous lemma to show by induction on i that $\widehat{\mathfrak{V}}(\Gamma_i) = \mathbf{t}$ for all natural numbers i with $0 \leq i \leq n$.

Theorem 132 (Correctness of PSC)

Any finite sequence Γ *of formulas in negation normal form which is derivable in* PSC *is valid; i.e.*

$$\vdash_{\mathsf{PSC}} \Gamma \implies \vDash \Gamma.$$

According to the previous definition, proofs in PSC are certain finite sequences of finite sequences of formulas in negation normal form. Alternatively, we can represent derivations in PSC as so-called *derivation trees*. These are rooted trees whose vertices are labeled with finite sequences of formulas and which are constructed according to the starting rules and extension rules of PSC introduced below. A derivation tree d with Γ associated to its root is often depicted as

$$\begin{array}{c} \vdots \\ d \\ \vdots \\ \Gamma \end{array}$$

Starting rules of PSC. For all finite sequences Γ, Δ, Π of formulas in negation normal form and all formulas A which are an atomic proposition or the complement of an atomic proposition: Any rooted tree which consists of its root only and has

$$\Gamma, \top, \Delta \qquad \text{or} \qquad \Gamma, A, \Delta, \overline{A}, \Pi$$

associated to its root is a derivation tree of PSC.

Extension rules of PSC. For all finite sequences Γ, Δ of formulas in negation normal form and all formulas A, B in negation normal form:

- A derivation tree

$$\begin{array}{c} \vdots \\ d \\ \vdots \\ \Gamma, A, B, \Delta \end{array}$$

is extended to the derivation tree with new root

$$\dfrac{\begin{array}{c} \vdots \\ d \\ \vdots \\ \Gamma, A, B, \Delta \end{array}}{\Gamma, A \vee B, \Delta}\,.$$

- Derivation trees

$$\begin{array}{cc} \vdots & \vdots \\ d_1 & d_2 \\ \vdots & \vdots \\ \Gamma, A, \Delta & \Gamma, B, \Delta \end{array}$$

are extended to the derivation tree with new root

$$\dfrac{\begin{array}{cc} \vdots & \vdots \\ d_1 & d_2 \\ \vdots & \vdots \\ \Gamma, A, \Delta & \Gamma, B, \Delta \end{array}}{\Gamma, A \wedge B, \Delta}\,.$$

As one can easily see, the starting rules of PSC correspond to the axioms of PSC and the extension rules of PSC to the rules of inference of PSC.

Example 133 Consider the following derivation tree d in PSC:

$$\dfrac{\dfrac{\dfrac{\overline{P}, \overline{P} \wedge \overline{R}, P, Q \qquad \overline{Q}, \overline{P} \wedge \overline{R}, P, Q}{\overline{P} \wedge \overline{Q}, \overline{P} \wedge \overline{R}, P, Q} \qquad \dfrac{\overline{P} \wedge \overline{Q}, \overline{P}, P, R \qquad \overline{P} \wedge \overline{Q}, \overline{R}, P, R}{\overline{P} \wedge \overline{Q}, \overline{P} \wedge \overline{R}, P, R}}{\overline{P} \wedge \overline{Q}, \overline{P} \wedge \overline{R}, P, Q \wedge R}}{\overline{P} \wedge \overline{Q}, \overline{P} \wedge \overline{R}, P \vee (Q \wedge R)}$$

This derivation tree d can be extended to a derivation tree e as follows:

$$\dfrac{\dfrac{\dfrac{\vdots}{d}\vdots}{(\overline{P}\wedge\overline{Q}),(\overline{P}\wedge\overline{R}),P\vee(Q\wedge R)}}{(\overline{P}\wedge\overline{Q})\vee(\overline{P}\wedge\overline{R}),P\vee(Q\wedge R)}}{((\overline{P}\wedge\overline{Q})\vee(\overline{P}\wedge\overline{R}))\vee(P\vee(Q\wedge R))}$$

The formula which labels the root of e is according to the definition of complements the formula

$$\overline{(P\vee Q)\wedge(P\vee R)}\vee(P\vee(Q\wedge R))$$

and as such equivalent to

$$((P\vee Q)\wedge(P\vee R))\to(P\vee(Q\wedge R)).$$

Hence, modulo transformation into negation normal form, this formula has been derived in PSC by means of derivation trees.

So we have seen that proofs in PSC can be understood as finite lists in the sense of Definition 130 or as derivations trees as above. There is even a third form which explicitly mentions a specific bound.

Definition 134 *Let Γ be a finite sequence of formulas in negation normal form. Then we define $\vdash^n_{\mathsf{PSC}}\Gamma$ for all natural numbers n by induction on n as follows:*

1. *If Γ is an axiom of PSC, then we have $\vdash^n_{\mathsf{PSC}}\Gamma$ for all natural numbers n.*
2. *If $\vdash^{n_i}_{\mathsf{PSC}}\Gamma_i$ and $n_i<n$ for every premise Γ_i of a rule of inference of PSC, then we have $\vdash^n_{\mathsf{PSC}}\Gamma$ for the conclusion Γ of this rule.*

Hence $\vdash^n_{\mathsf{PSC}}\Gamma$ means that Γ is provable in PSC by a derivation tree whose depth is bounded by n. Clearly, we have $\vdash_{\mathsf{PSC}}\Gamma$ in the sense of Definition 130 if and only if there exists a natural number n such that $\vdash^n_{\mathsf{PSC}}\Gamma$. In the following lemma we summarize a few important structural properties of provability in PSC.

Lemma 135 (Structural lemma)

Let Γ,Δ be finite sequences of formulas in negation normal form. Then we have for all natural numbers m,n:

1. $\vdash^m_{\mathsf{PSC}}\Gamma$ *and* Δ *is a permutation of* $\Gamma \implies \vdash^m_{\mathsf{PSC}}\Delta$.
2. $\vdash^m_{\mathsf{PSC}}\Gamma$ *and* $m\leq n \implies \vdash^n_{\mathsf{PSC}}\Gamma$.
3. $\vdash^m_{\mathsf{PSC}}\Gamma \implies \vdash^m_{\mathsf{PSC}}\Gamma,\Delta$.

PROOF The first and the third assertion are established by complete induction on m; the second assertion follows directly from the definition above. □

Lemma 136 (Inversion lemma)

Let Γ be finite sequence of formulas in negation normal form and let A, B be formulas in negation normal form. Then we have for all natural numbers n:

1. $\vdash^n_{\mathsf{PSC}} \Gamma, A \vee B \implies \vdash^n_{\mathsf{PSC}} \Gamma, A, B.$
2. $\vdash^n_{\mathsf{PSC}} \Gamma, A \wedge B \implies \vdash^n_{\mathsf{PSC}} \Gamma, A$ *and* $\vdash^n_{\mathsf{PSC}} \Gamma, B.$

PROOF The proof of both assertions is by complete induction on n. We confine ourselves to discussing the first assertion; the second is treated accordingly.

For the proof of the first assertion we distinguish the following cases:

(i) $\Gamma, A \vee B$ is an axiom of PSC. Then our assertion is trivially satisfied.

(ii) The last inference in the derivation of $\Gamma, A \vee B$ has been of the form

$$(\vee) \qquad \frac{\Gamma, A, B}{\Gamma, A \vee B}.$$

Then there exists a natural number $n_0 < n$ such that $\vdash^{n_0}_{\mathsf{PSC}} \Gamma, A, B$. According to the second part of the previous lemma we thus have $\vdash^n_{\mathsf{PSC}} \Gamma, A, B$.

(iii) In all other cases we simply apply the induction hypothesis to the premise(s) of the last inference and carry through this inference afterwards, yielding what we want. □

Lemma 137 (Contraction lemma)

Let Γ be finite sequence of formulas in negation normal form and let A be a formula in negation normal form. Then we have for all natural numbers n that

$$\vdash^n_{\mathsf{PSC}} \Gamma, A, A \implies \vdash^n_{\mathsf{PSC}} \Gamma, A.$$

PROOF Again we proceed by complete induction on n and distinguish the following cases:

(i) Γ, A, A is an axiom of PSC. Then Γ, A is an axiom of PSC as well and the assertion is clear.

(ii) A is of the form $B \vee C$ and the last inference in the derivation of Γ, A, A has been of the form

$$(\vee) \qquad \frac{\Gamma, B, C, B \vee C}{\Gamma, B \vee C, B \vee C}.$$

Then there exists a natural number $n_0 < n$ such that

$$\vdash^{n_0}_{\mathsf{PSC}} \Gamma, B, C, B \vee C.$$

In view of Lemma 136.1 we thus also have

$$\vdash^{n_0}_{\mathsf{PSC}} \Gamma, B, C, B, C.$$

According to Lemma 135.1 we can transform this into

$$\vdash^{n_0}_{\mathsf{PSC}} \Gamma, C, C, B, B$$

such that the induction hypothesis yields

$$\vdash^{n_0}_{\mathsf{PSC}} \Gamma, C, C, B.$$

Applying Lemma 135.1 once more gives us

$$\vdash^{n_0}_{\mathsf{PSC}} \Gamma, B, C, C,$$

from which the induction hypothesis implies

$$\vdash^{n_0}_{\mathsf{PSC}} \Gamma, B, C.$$

Making use of the disjunction rule $(\vee)$, we therefore obtain

$$\vdash^{n}_{\mathsf{PSC}} \Gamma, B \vee C,$$

as desired.

(iii) A is of the form $B \vee C$ and the last inference in the derivation of Γ, A, A has been of the form

$$(\vee) \qquad \frac{\Gamma, B \vee C, B, C}{\Gamma, B \vee C, B \vee C}.$$

Similar to the previous case.

(iv) A is of the form $B \wedge C$ and the last inference in the derivation of Γ, A, A has been of the form

$$(\wedge) \qquad \frac{\Gamma, B, B \wedge C \qquad \Gamma, C, B \wedge C}{\Gamma, B \wedge C, B \wedge C}.$$

Similar to case (ii).

(v) A is of the form $B \wedge C$ and the last inference in the derivation of Γ, A, A has been of the form

$$(\wedge) \qquad \frac{\Gamma,\, B \wedge C,\, B \qquad \Gamma,\, B \wedge C,\, C}{\Gamma,\, B \wedge C,\, B \wedge C}.$$

Similar to case (ii).

(vi) In all other cases we simply apply the induction hypothesis to the premise(s) of the last inference and carry through this inference afterwards, yielding what we want. □

Theorem 138 *If Γ and Δ are finite sequences of formulas in negation normal form, then we have for all natural numbers n that*

$$\vdash^{n}_{\mathsf{PSC}} \Gamma \;\; \textit{and} \;\; \mathrm{set}(\Gamma) = \mathrm{set}(\Delta) \quad \Longrightarrow \quad \vdash^{n}_{\mathsf{PSC}} \Delta.$$

PROOF This theorem is an immediate consequence of Lemma 135.1, Lemma 135.3, and the previous lemma. □

2.8 Deduction chains for PSC

In this section we present Schütte's method of deduction chains (see Schütte [11]) and use it in order to show that our calculus PSC is complete. Before presenting the central concept, we need some preparatory definitions.

Definition 139

1. *A formula A in negation normal form is called* irreducible *if it is an atomic proposition, the negation of an atomic proposition, or a propositional constant; all other formulas in negation normal form are called* reducible.

2. *A finite sequence of formulas in negation normal form which contains at least one reducible formula is called* reducible*; all other finite sequences of formulas in negation normal form are called* irreducible.

3. *The right most reducible formula of a reducible sequence of formulas in negation normal form is called the* distinguished *formula of this sequence.*

The basic idea in forming deduction chains is to proceed inversely to the inference rules of PSC, and to do this in a systematic way.

Definition 140 *Let Γ be a finite sequence of formulas in negation normal form. A* deduction chain *– D-chain for short – for Γ is a finite sequence*

$$\Gamma_1, \Gamma_2, \ldots$$

of finite sequences of formulas in negation normal form generated as follows:

(D.1) *The initial sequence Γ_1 of the D-chain is the sequence Γ.*

(D.2) *If the sequence Γ_n of the D-chain is irreducible or an axiom of* PSC*, then it is the last sequence of this D-chain. We then say that the D-chain has length n.*

(D.3) *If the sequence Γ_n of the D-chain is reducible and not an axiom of* PSC*, then Γ_n has an immediate successor Γ_{n+1} in this D-chain which is determined as follows. We first write Γ_n as*

$$\Pi_n, A, \Sigma_n,$$

where A is the distinguished formula of Γ_n, and distinguish the following two cases:

(D.3.1) *A is a formula $B \vee C$. Then Γ_{n+1} is the sequence*

$$\Pi_n, B, C, \Sigma_n.$$

(D.3.2) *A is a formula $B \wedge C$. Then Γ_{n+1} is either the sequence*

$$\Pi_n, B, \Sigma_n \qquad \textit{or the sequence} \qquad \Pi_n, C, \Sigma_n.$$

Clearly, when building up the D-chains of a finite sequence of formulas in negation normal forms, we will always reach, after finitely many steps, a sequence which is irreducible or an axiom of PSC. Furthermore, there are only finitely many D-chains which begin with the same finite sequence of formulas in negation normal form.

The completeness result for PSC will be an immediate consequence of the following principal syntactic lemma and principal semantic lemma.

Lemma 141 (Principal syntactic lemma)

Let Γ be a finite sequence of formulas in negation normal form. If every D-chain for Γ ends with an axiom of PSC*, then we have $\vdash_{\mathsf{PSC}} \Gamma$.*

PROOF The basic idea of this proof is very simple. If all D-chains for Γ end with an axiom of PSC, then it is possible to arrange these D-chains so that they form a derivation tree in PSC. More formally: Since there are only finitely many D-chains for Γ, there exists a natural number m which is the maximal length of these D-chains. By complete induction on n we show for any n, $0 \leq n \leq m-1$:

If Δ occurs at position $(m-n)$ of a D-chain for Γ, then $\vdash_{\mathsf{PSC}} \Delta$.

Now we set $n = m-1$ in this auxiliary assertion and obtain $\vdash_{\mathsf{PSC}} \Gamma_1$. Since Γ_1 is the sequence Γ we have proved what we want. □

Lemma 142 (Principal semantic lemma)

Let Γ be a finite sequence of formulas in negation normal form. If there exists a D-chain for Γ which does not end with an axiom of PSC*, then there exists a valuation $\mathfrak{V}$ such that $\widehat{\mathfrak{V}}(\Gamma) = \mathbf{f}$.*

PROOF Assume that

$$\Gamma_1, \Gamma_2, \ldots, \Gamma_k$$

is a D-chain for Γ such that none of the Γ_i, $1 \leq i \leq k$ is an axiom of PSC and Γ_k is irreducible. Now let K be the collection of all formulas which occur in one of the $\Gamma_1, \Gamma_2, \ldots, \Gamma_k$,

$$K \;:=\; \mathrm{set}(\Gamma_1) \cup \mathrm{set}(\Gamma_2) \cup \ldots \cup \mathrm{set}(\Gamma_k).$$

This D-chain and the collection K then have the following properties:

(1) If an atomic proposition, the negation of an atomic proposition, or a propositional constant occurs in Γ_m, then it also occurs in all Γ_n for $m \leq n$.

(2) There is no atomic proposition P such that $P \in K$ and $\neg P \in K$; in addition, $\top \notin K$.

(3) If a reducible A belongs to K, then there exists an n, $1 \leq n < k$, such that A is the distinguished formula of Γ_n.

(4) If a formula $A \vee B$ belongs to K, then A and B are elements of K as well.

(5) If a formula $A \wedge B$ belongs to K, then A or B is an element of K as well.

Based on K we now define a valuation $\mathfrak{V}$ by setting, for any atomic proposition P,

$$\mathfrak{V}(P) \;:=\; \begin{cases} \mathbf{f} & \text{if } P \text{ is an element of } K, \\ \mathbf{t} & \text{if } P \text{ does not belong to } K. \end{cases}$$

The next step is to show, by induction on $\ell(A)$, that for all formulas A

$$(*) \qquad A \in K \implies \widehat{\mathfrak{V}}(A) = \mathbf{f}.$$

To do so, we distinguish the following cases:

(i) A is an atomic proposition. Then the assertion is immediate from the definition of $\mathfrak{V}$.

(ii) A is the negation $\neg P$ of an atomic proposition P. Because of (2) we then know that $P \notin K$. Hence $\mathfrak{V}(P) = \mathbf{t}$, implying that $\widehat{\mathfrak{V}}(A) = \widehat{\mathfrak{V}}(\neg P) = \mathbf{f}$.

(iii) A is a propositional constant. Then (2) implies that A is the constant $\bot$, i.e. $\widehat{\mathfrak{V}}(A) = \mathbf{f}$.

(iv) A is a formula $B \vee C$. Then (4) yields $B \in K$ and $C \in K$. Hence the induction hypothesis implies $\widehat{\mathfrak{V}}(B) = \mathbf{f}$ and $\widehat{\mathfrak{V}}(C) = \mathbf{f}$. As an immediate consequence we have $\widehat{\mathfrak{V}}(A) = \widehat{\mathfrak{V}}(B \vee C) = \mathbf{f}$.

(v) A is a formula $B \wedge C$. Then (5) yields $B \in K$ or $C \in K$. Hence the induction hypothesis implies $\widehat{\mathfrak{V}}(B) = \mathbf{f}$ or $\widehat{\mathfrak{V}}(C) = \mathbf{f}$. As an immediate consequence we have in this case that $\widehat{\mathfrak{V}}(A) = \widehat{\mathfrak{V}}(B \wedge C) = \mathbf{f}$.

This finishes the proof of (*). Since $\mathrm{set}(\Gamma) \subseteq K$, we thus have $\widehat{\mathfrak{V}}(A) = \mathbf{f}$ for all $A \in \mathrm{set}(\Gamma)$. However, this directly gives us $\widehat{\mathfrak{V}}(\Gamma) = \mathbf{f}$, as desired. □

Theorem 143 (Completeness of PSC)

Any finite sequence Γ of formulas in negation normal form which is valid is derivable in PSC*; i.e.*

$$\vDash \Gamma \implies \vdash_{\mathsf{PSC}} \Gamma.$$

Proof Let Γ be a finite sequence of formulas in negation normal form which is valid. Then the principal semantic lemma implies that any D-chain for Γ ends with an axiom of PSC. Hence the principal syntactic lemma yields $\vdash_{\mathsf{PSC}} \Gamma$. □

Corollary 144 *For any finite sequence Γ of formulas in negation normal form we have*

$$\vdash_{\mathsf{PSC}} \Gamma \iff \vDash \Gamma.$$

Proof This equivalence is, of course, immediate from the correctness and completeness of PSC. □

2.9 Adding theories to PSC

In the previous section we were concerned with pure logic and testing for validity. Now we extend our approach via deduction chains in a way that also theories can be dealt with.

Theorem 145 (Correctness of PSC with respect to theories)
Let T be a theory consisting of formulas in negation normal form and let Γ be a finite sequence of formulas in negation normal form. For all $A_1, \ldots, A_n \in T$ we then have that

$$\vdash_{\mathsf{PSC}} \overline{A_1}, \ldots, \overline{A_n}, \Gamma \quad \Longrightarrow \quad T \vDash \Gamma.$$

PROOF If Γ is the sequence of formulas $B_1, \ldots, B_m$, then Theorem 132 implies

$$\vDash \overline{A_1} \vee \ldots \vee \overline{A_n} \vee B_1 \vee \ldots \vee B_m.$$

Since $T \vDash A_i$ for $1 \leq i \leq n$, we immediately obtain our assertion. □

Until the end of this section we assume that T is a non-empty theory consisting of formulas in negation normal form and that

$$F_1, F_2, \ldots$$

is an arbitrary but fixed infinite list of formulas which consists exactly of the elements of T. If T is finite, then this list must contain some elements of T infinitely often.

Definition 146 *Let Γ be a finite sequence of formulas in negation normal form. A* deduction chain *– D-chain for short – for Γ* with respect to *T is a sequence*

$$\Gamma_1, \Gamma_2, \ldots$$

of finite sequences of formulas in negation normal form generated as follows:

($D_T.1$) *The initial sequence Γ_1 of the D-chain is the sequence $\overline{F_1}, \Gamma$.*

($D_T.2$) *If the sequence Γ_n of the D-chain is an axiom of PSC, then it is the last sequence of this D-chain. We then say that the D-chain has length n.*

($D_T.3$) *If the sequence Γ_n of the D-chain is irreducible and not an axiom of PSC, then the immediate successor Γ_{n+1} in this D-chain of Γ_n is the sequence*

$$\overline{F_{n+1}}, \Gamma_n.$$

(D_T.4) *If the sequence Γ_n of the D-chain is reducible and not an axiom of* PSC*, then Γ_n has an immediate successor Γ_{n+1} in this D-chain which is determined as follows. We first write Γ_n as*

$$\Pi_n,\, A,\, \Sigma_n,$$

where A is the distinguished formula of Γ_n, and distinguish the following two cases:

(D_T.4.1) *A is a formula $B \vee C$. Then Γ_{n+1} is the sequence*

$$\overline{F_{n+1}},\, \Pi_n,\, B,\, C,\, \Sigma_n.$$

(D_T.4.2) *A is a formula $B \wedge C$. Then Γ_{n+1} is either the sequence*

$$\overline{F_{n+1}},\, \Pi_n,\, B,\, \Sigma_n \qquad \textit{or the sequence} \qquad \overline{F_{n+1}},\, \Pi_n,\, C,\, \Sigma_n.$$

As in the previous section we are now going to prove a principal syntactic and a principal semantic lemma and deduce our completeness result with respect to theories from those.

Lemma 147 (Principal syntactic lemma with respect to theory T)
Let Γ be a finite sequence of formulas in negation normal form. If every D-chain for Γ with respect to T is finite, then there exist $A_1, \ldots, A_n \in T$ such that

$$\vdash_{\mathsf{PSC}} \overline{A_1},\, \ldots,\, \overline{A_n},\, \Gamma.$$

PROOF We may assume that all these D-chains are arranged as a rooted tree. Then this tree is finitely branching, and all its branches are finite. In view of König's lemma this tree has thus to be finite. Accordingly, there are only finitely many D-chains for Γ with respect to T. Let m be the maximal length of these D-chains. By complete induction on n we now show for any n, $0 \leq n \leq m-1$:

If Δ occurs at position $(m-n)$ of a D-chain for Γ, then

$$\begin{cases} \vdash_{\mathsf{PSC}} \Delta & \text{if } n = 0, \\ \vdash_{\mathsf{PSC}} \overline{F_m},\, \ldots,\, \overline{F_{m-(n-1)}},\, \Delta & \text{if } 0 < n < m. \end{cases}$$

Now we set $n = m-1$ in this auxiliary assertion and obtain

$$\vdash_{\mathsf{PSC}} \overline{F_m},\, \ldots,\, \overline{F_2},\, \Gamma_1.$$

Since Γ_1 is the sequence $\overline{F_1}, \Gamma$ we have proved what we want if we choose the formulas $A_1, \ldots, A_n$ such that $\{A_1, \ldots, A_n\} = \{F_1, \ldots, F_m\}$. □

Lemma 148 (Principal semantic lemma with respect to theory T)
Let Γ be a finite sequence of formulas in negation normal form. If there exists an infinite D-chain for Γ with respect to T, then there exists a valuation $\mathfrak{V}$ such that the following two properties are satisfied:

1. $\widehat{\mathfrak{V}}(A) = \mathbf{t}$ *for all elements A of T.*
2. $\widehat{\mathfrak{V}}(\Gamma) = \mathbf{f}$.

PROOF In principle, this proof is identical to the proof of Lemma 142. We pick our infinite D-chain

$$\Gamma_1, \Gamma_2, \ldots$$

for Γ with respect to T. As above we set

$$K := \bigcup_{i=1}^{\infty} \mathrm{set}(\Gamma_i)$$

and obtain for K and this D-chain the same properties (1)–(5) as in the proof of Lemma 142. The valuation $\mathfrak{V}$ is also defined as above, by setting, for any atomic proposition P,

$$\mathfrak{V}(P) := \begin{cases} \mathbf{f} & \text{if } P \text{ is an element of } K, \\ \mathbf{t} & \text{if } P \text{ does not belong to } K, \end{cases}$$

and again we obtain that for all formulas A

$$(*) \qquad A \in K \implies \widehat{\mathfrak{V}}(A) = \mathbf{f}.$$

However, by construction of our D-chain we also know that $\overline{F_i} \in K$ for any positive natural number i, implying that $\widehat{\mathfrak{V}}(A) = \mathbf{t}$ for any $A \in T$. Furthermore, $\widehat{\mathfrak{V}}(\Gamma) = \mathbf{f}$ follows from $\mathrm{set}(\Gamma) \subseteq K$. □

Theorem 149 (Completeness of PSC with respect to theory T)
Let Γ be a finite sequence of formulas in negation normal form. If $T \models \Gamma^{\vee}$, then there exist $A_1, \ldots, A_n \in T$ such that

$$\vdash_{\mathsf{PSC}} \overline{A_1}, \ldots, \overline{A_n}, \Gamma.$$

PROOF Let Γ be a finite sequence of formulas in negation normal form which follows from T. Then the principal semantic lemma implies that any D-chain for Γ with respect to T has to be finite. Hence the principal syntactic lemma yields our assertion. □

2.10 Resolution

Resolution is a further form of inference leading to a correct and complete procedure for testing whether a formula follows from a theory. It is a powerful method, based on one simple rule, leading to a search space which is much smaller than that of many other approaches. There exist many forms and variants of resolution: unit resolution, input resolution, and SLD resolution, to mention a few. Resolution was introduced by John Alan Robinson in 1965, see Robinson [10]. For a detailed exposition see, for example, Leitsch [7].

Resolution works on so-called clause sets. After introducing some basic terminology, we first show that all formulas are satisfiability equivalent to clause sets. Afterwards we present the most simple and general form of resolution.

Definition 150

1. *A* literal *is an atomic proposition or the negation of an atomic proposition; atomic propositions are called* positive literals *and their negations* negative literals*.*

2. *A* clause *is a finite set of literals; the empty clause is often denoted by* $\Box$*.*

3. *Given a valuation* $\mathfrak{V}$ *and the clause* $\Phi = \{L_1, \ldots, L_n\}$*, we set*

$$\widehat{\mathfrak{V}}(\Phi) := \begin{cases} \mathbf{t} & \text{if } \widehat{\mathfrak{V}}(L_i) = \mathbf{t} \text{ for some } i,\ 1 \leq i \leq n, \\ \mathbf{f} & \text{if } \widehat{\mathfrak{V}}(L_i) = \mathbf{f} \text{ for all } i,\ 1 \leq i \leq n. \end{cases}$$

4. *A* clause set *is a finite or infinite set of clauses.*

5. *A valuation* $\mathfrak{V}$ satisfies *a clause set* $\mathbb{K}$ *if and only if* $\widehat{\mathfrak{V}}(\Phi) = \mathbf{t}$ *for all elements* Φ *of* $\mathbb{K}$*.*

6. *A clause set* $\mathbb{K}$ *is* satisfiable *if and only if there exists a valuation which satisfies* $\mathbb{K}$*.*

7. *Given a valuation* $\mathfrak{V}$ *and the clause set* $\mathbb{K} = \{\Phi_1, \ldots, \Phi_n\}$*, we set*

$$\widehat{\mathfrak{V}}[\mathbb{K}] := \begin{cases} \mathbf{t} & \text{if } \widehat{\mathfrak{V}}(\Phi_i) = \mathbf{t} \text{ for all } i,\ 1 \leq i \leq n, \\ \mathbf{f} & \text{if } \widehat{\mathfrak{V}}(\Phi_i) = \mathbf{f} \text{ for some } i,\ 1 \leq i \leq n. \end{cases}$$

8. *A clause set* $\mathbb{K}$ *is called* logically equivalent *to a formula* A *if and only if* $\widehat{\mathfrak{V}}[\mathbb{K}] = \widehat{\mathfrak{V}}(A)$ *for all valuations* $\mathfrak{V}$*.*

If a valuation $\mathfrak{V}$ is applied to a set of clauses $\mathbb{K}$, we write $\widehat{\mathfrak{V}}[\mathbb{K}]$ rather than $\widehat{\mathfrak{V}}(\mathbb{K})$ in order to avoid a notational conflict in the case of the empty set which can be both: the empty clause and the empty clause set.

For notational convenience we use as additional syntactic variables (metavariables), possibly with subscripts or primed:

K, L, M	for literals,
Υ, Φ, Ψ	for clauses,
$\mathbb{K}, \mathbb{L}$	for clause sets.

Remark 151

1. Given a valuation $\mathfrak{V}$, we have $\widehat{\mathfrak{V}}(\square) = \mathbf{f}$ and $\widehat{\mathfrak{V}}[\varnothing] = \mathbf{t}$.
2. The empty clause set $\varnothing$ is logically equivalent to the formula $\top$.
3. The clause set $\{\square\}$ is logically equivalent to the formula $\bot$.

In order to show that any formula is logically equivalent to a clause set, we first convince ourselves that any formula can be brought into conjunctive normal form, where a formula is said to be in conjunctive normal form if it is a conjunction of disjunctions of literals.

Definition 152 *The formulas in* conjunctive normal form *are all formulas of the form*

$$\bigwedge_{i=1}^{m} (L_{i,1} \vee \ldots \vee L_{i,n_i})$$

for natural numbers $m, n_1, \ldots, n_m$.

Example 153

1. The following formulas are in conjunctive normal form:

$$P, \quad \neg P, \quad P_1 \vee P_2 \vee P_3, \quad P_1 \wedge P_2 \wedge P_3, \quad (P_1 \vee Q_1) \wedge (P_2 \vee Q_2 \vee Q_3) \wedge P_3.$$

2. The following formulas are not in conjunctive normal form:

$$\top, \quad \neg\neg P, \quad P \to Q, \quad \neg(P \vee Q), \quad P \vee (Q \wedge R).$$

It is obvious from this definition that a formula in conjunctive normal form is in negation normal form. However, in contrast to negation normal forms, formulas in conjunctive normal form must not contain propositional constants.

In order to transform a formula A into an equivalent formula in conjunctive normal form, we proceed as follows: (i) remove all propositional constants from A; (ii) translate the resulting formula A' into its negation normal form $\mathrm{nnf}(A')$; (iii) use the equivalences

$$B_1 \vee (B_2 \wedge B_3) \quad \text{equivalent to} \quad (B_1 \vee B_2) \wedge (B_1 \vee B_3),$$
$$(B_2 \wedge B_3) \vee B_1 \quad \text{equivalent to} \quad (B_2 \vee B_1) \wedge (B_3 \vee B_1).$$

to exchange disjunctions with conjunctions. The formally exact definition is as follows.

Definition 154

1. *Given a formula A, let $\sigma_0(A)$ be the formula obtained from A by replacing all propositional constants $\top$ by $(\mathfrak{p}_0 \vee \neg\mathfrak{p}_0)$ and all propositional constants $\bot$ by $(\mathfrak{p}_0 \wedge \neg\mathfrak{p}_0)$, respectively. Then set $\sigma(A) := \mathrm{nnf}(\sigma_0(A))$.*

2. *The CNF-reduction $\tau(A)$ of a formula A in negation normal form without propositional constants is obtained by, starting with A, iteratively replacing all subformulas of the form*

$$D_1 \vee (D_2 \wedge D_3) \quad \textit{by} \quad (D_1 \vee D_2) \wedge (D_1 \vee D_3),$$
$$(D_2 \wedge D_3) \vee D_1 \quad \textit{by} \quad (D_2 \vee D_1) \wedge (D_3 \vee D_1).$$

3. *Finally, for any formula A we set*

$$\mathrm{cnf}(A) \;:=\; \tau(\sigma(A)).$$

Given this translation of a formula A into the formula $\mathrm{cnf}(A)$, the following lemma should be obvious; all details can be easily checked by the reader.

Lemma 155 *For any formula A, its translation $\mathrm{cnf}(A)$ is equivalent to A and in conjunctive normal form. In addition, $\mathrm{Atp}(\mathrm{cnf}(A)) \subseteq \mathrm{Atp}(A) \cup \{\mathfrak{p}_0\}$.*

Example 156 Let A be the formula $(P \wedge \bot) \vee \neg(Q \vee R)$. Then A is successively transformed into the following formulas (some intermediate steps are left out):

- $(P \wedge (\mathfrak{p}_0 \wedge \neg\mathfrak{p}_0)) \vee \neg(Q \vee R)$,

- $(P \wedge (\mathfrak{p}_0 \wedge \neg\mathfrak{p}_0)) \vee (\neg Q \wedge \neg R)$,
- $((P \wedge (\mathfrak{p}_0 \wedge \neg\mathfrak{p}_0)) \vee \neg Q) \wedge ((P \wedge (\mathfrak{p}_0 \wedge \neg\mathfrak{p}_0)) \vee \neg R)$,
- $(P \vee \neg Q) \wedge (\mathfrak{p}_0 \vee \neg Q) \wedge (\neg\mathfrak{p}_0 \vee \neg Q) \wedge (P \vee \neg R) \wedge (\mathfrak{p}_0 \vee \neg R) \wedge (\neg\mathfrak{p}_0 \vee \neg R)$.

Lemma 157 *For any formula A there exists a clause set $\mathbb{K}$ which is logically equivalent to A.*

PROOF We transform A into conjunctive normal form $\mathrm{cnf}(A)$, which can be written as

$$\bigwedge_{i=1}^{m} (L_{i,1} \vee \ldots \vee L_{i,n_i})$$

for suitable natural numbers $m, n_1, \ldots, n_m$ and literals $L_{1,1}, \ldots, L_{m,n_m}$. Then we simply set

$$\mathbb{K} \coloneqq \{\, \{L_{i,1}, \ldots, L_{i,n_i}\} : 1 \leq i \leq m \,\}.$$

It is immediate from the previous considerations that this clause set $\mathbb{K}$ is logically equivalent to A. □

Remark 158 The time complexity for determining the conjunctive normal form $\mathrm{cnf}(A)$ of a formula A according to this procedure may be exponential in the length of A. Pick atomic propositions $P_1, \ldots, P_n, Q_1, \ldots, Q_n$ and consider the formulas

$$A_n \coloneqq (P_1 \wedge Q_1) \vee \ldots \vee (P_n \wedge Q_n).$$

Then $\ell(A_n) = 2n + 2n - 1$. Modulo omitting minor details, our procedure translates any A_n into

$$A_n' \coloneqq (P_1 \vee \ldots P_{n-1} \vee P_n) \wedge (P_1 \vee \ldots P_{n-1} \vee Q_n) \wedge \ldots \wedge (Q_1 \vee \ldots Q_{n-1} \vee Q_n)$$

with 2^n disjunctions; each disjunction contains either P_i or Q_i for each $1 \leq i \leq n$. Clearly, $2^n \leq \ell(A_n')$, and our transformation of A_n into A_n' requires more than 2^n steps. Hence the time complexity of the transformation of A_n into $\mathrm{cnf}(A_n)$ and the length of $\mathrm{cnf}(A_n)$ are exponential in n and thus in the length of A_n.

This remark tells us that the transformation of a formula into an equivalent clause set is problematic from the point of view of computational complexity. Fortunately, however, there is a more efficient procedure which associates to any formula a clause set preserving satisfiability rather than logical equivalence.

Definition 159 *Let A be a formula in negation normal form without propositional constants.*

1. *For any element B of* $\mathrm{Sufo}(A)$ *we choose a new atomic proposition Q_B such that the following two conditions are satisfied:*
   (a) *Q_B does not occur in A.*
   (b) *Q_{B_1} and Q_{B_2} are different for any different elements $B_1, B_2 \in \mathrm{Sufo}(A)$.*
2. *Depending on this association of atomic propositions to the subformulas of A a finite clause set $\mathbb{K}_A(B)$ is inductively defined for any $B \in Sufo(A)$ as follows:*
   (a) *If B is a literal, then*
   $$\mathbb{K}_A(B) := \{\, \{\overline{B}, Q_B\},\, \{\neg Q_B, B\} \,\}.$$
   (b) *If B is a formula $C \vee D$, then*
   $$\mathbb{K}_A(B) := \begin{cases} \mathbb{K}_A(C) \cup \mathbb{K}_A(D) \cup \\ \{\, \{\neg Q_C, Q_B\},\, \{\neg Q_D, Q_B\},\, \{\neg Q_B, Q_C, Q_D\} \,\}. \end{cases}$$
   (c) *If B is a formula $C \wedge D$, then*
   $$\mathbb{K}_A(B) := \begin{cases} \mathbb{K}_A(C) \cup \mathbb{K}_A(D) \cup \\ \{\, \{\neg Q_C, \neg Q_D, Q_B\},\, \{\neg Q_B, Q_C\},\, \{\neg Q_B, Q_D\} \,\}. \end{cases}$$
3. *Finally, we set $\mathbb{K}_A := \mathbb{K}_A(A)$.*

By our choice of the new atomic propositions we make sure that every subformula B of A is linked with a new atomic proposition Q_B. The clauses associated to Q_B express that Q_B is equivalent to B.

Example 160 Let A be the formula $P_0 \vee (P_1 \wedge P_2)$. Then $\mathrm{Sufo}(A)$ contains exactly the following elements:

$$P_0, \quad P_1, \quad P_2, \quad P_1 \wedge P_2, \quad P_0 \vee (P_1 \wedge P_2).$$

Hence we need the following new atomic propositions:

$$Q_{P_0}, \quad Q_{P_1}, \quad Q_{P_2}, \quad Q_{P_1 \wedge P_2}, \quad Q_{P_0 \vee (P_1 \wedge P_2)}.$$

In addition, we have that $\mathbb{K}_A$ is the clause set

$$\begin{aligned} &\{\, \{\neg P_0, Q_{P_0}\},\ \{\neg Q_{P_0}, P_0\},\ \{\neg P_1, Q_{P_1}\},\ \{\neg Q_{P_1}, P_1\},\ \{\neg P_2, Q_{P_2}\},\ \{\neg Q_{P_2}, P_2\}, \\ &\ \{\neg Q_{P_1}, \neg Q_{P_2}, Q_{P_1 \wedge P_2}\},\ \{\neg Q_{P_1 \wedge P_2}, Q_{P_1}\},\ \{\neg Q_{P_1 \wedge P_2}, Q_{P_2}\},\ \{\neg Q_{P_0}, Q_{P_0 \vee (P_1 \wedge P_2)}\}, \\ &\ \{\neg Q_{P_1 \wedge P_2}, Q_{P_0 \vee (P_1 \wedge P_2)}\},\ \{\neg Q_{P_0 \vee (P_1 \wedge P_2)}, Q_{P_0}, Q_{P_1 \wedge P_2}\} \,\}. \end{aligned}$$

The following lemma provides an upper bound of the sizes of the clause sets $\mathbb{K}_A(B)$ in terms of the lengths of the subformulas B of A. Its proof is by a more or less trivial induction on $\ell(B)$ and can be omitted.

Lemma 161 *Let A be a formula in negation normal form without propositional constants and choose $(Q_B : B \in \mathrm{Sufo}(A))$ according to Definition 159. Then we have for all $B \in \mathrm{Sufo}(A)$ that $|\mathbb{K}_A(B)| \leq 3 \cdot \ell(B)$; in particular, $|\mathbb{K}_A| \leq 3 \cdot \ell(A)$.*

Now we turn to showing that $\mathbb{K}_A$ is satisfiable if and only if A is satisfiable. This will be an easy consequence of the following two preparatory lemmas.

Lemma 162 *Let A be a formula in negation normal form without propositional constants and choose $(Q_B : B \in \mathrm{Sufo}(A))$ according to Definition 159. If $\mathfrak{V}$ is valuation which satisfies $\mathbb{K}_A$, then we have for all subformulas B of A that*

$$\widehat{\mathfrak{V}}(B) = \mathbf{t} \quad \Longleftrightarrow \quad \mathfrak{V}(Q_B) = \mathbf{t}.$$

Proof We show this assertion by induction on the length of the subformulas of A and distinguish the following cases:

(i) B is a literal. Then $\mathbb{K}_A(B)$, which is a subset of $\mathbb{K}_A$, contains the two clauses $\{\overline{B}, Q_B\}$ and $\{\neg Q_B, B\}$. Because of

$$\widehat{\mathfrak{V}}(\{\overline{B}, Q_B\}) \;=\; \widehat{\mathfrak{V}}(\{\neg Q_B, B\}) \;=\; \mathbf{t}$$

our assertion is obvious.

(ii) B is a formula $C \vee D$. According to the definition of $\mathbb{K}_A(B)$, which is a subset of $\mathbb{K}_A$, we know that $\mathfrak{V}$ satisfies the clause sets $\mathbb{K}_A(C)$ and $\mathbb{K}_A(D)$ as well as the clause set

$$\{\,\{\neg Q_C, Q_B\},\, \{\neg Q_D, Q_B\},\, \{\neg Q_B, Q_C, Q_D\}\,\}. \tag{1}$$

The induction hypothesis implies

$$\widehat{\mathfrak{V}}(C) = \mathbf{t} \quad \Longleftrightarrow \quad \mathfrak{V}(Q_C) = \mathbf{t}, \tag{2}$$

$$\widehat{\mathfrak{V}}(D) = \mathbf{t} \quad \Longleftrightarrow \quad \mathfrak{V}(Q_D) = \mathbf{t}. \tag{3}$$

If $\widehat{\mathfrak{V}}(B) = \mathbf{t}$, we have $\widehat{\mathfrak{V}}(C) = \mathbf{t}$ or $\widehat{\mathfrak{V}}(D) = \mathbf{t}$, and thus (2) and (3) imply that

$$\mathfrak{V}(Q_C) = \mathbf{t} \quad \text{or} \quad \mathfrak{V}(Q_D) = \mathbf{t}. \tag{4}$$

Since $\mathfrak{V}$ satisfies the clause set mentioned in (1), we also know that $\widehat{\mathfrak{V}}(\{\neg Q_C, Q_B\}) = \mathbf{t}$ and $\widehat{\mathfrak{V}}(\{\neg Q_D, Q_B\}) = \mathbf{t}$. Consequently, $\mathfrak{V}(Q_B) = \mathbf{t}$.

On the other hand, if $\mathfrak{V}(Q_B) = \mathbf{t}$, then the validity of the third clause of the clause set in (1) gives us (4). Together with (2) and (3) we immediately obtain $\widehat{\mathfrak{V}}(B) = \mathbf{t}$.

(iii) B is a formula $C \wedge D$. We simply proceed as in the previous case. □

Lemma 163 *Let A be a formula in negation normal form without propositional constants and choose $(Q_B : B \in \mathrm{Sufo}(A))$ according to Definition 159. For a given valuation $\mathfrak{V}$ we define the valuation $\mathfrak{W}$ by*

$$\mathfrak{W}(P) := \begin{cases} \mathfrak{V}(P) & \text{if } P \in \mathrm{Atp}(A), \\ \widehat{\mathfrak{V}}(B) & \text{if } P \text{ is the atomic proposition } Q_B, \\ \mathbf{f} & \text{otherwise}, \end{cases}$$

Then we have $\widehat{\mathfrak{W}}[\mathbb{K}_A(B)] = \mathbf{t}$ for all $B \in \mathrm{Sufo}(A)$.

Proof We first observe that the properties of $\mathfrak{W}$ imply that $\widehat{\mathfrak{W}}(B) = \widehat{\mathfrak{V}}(B)$ for all $B \in \mathrm{Sufo}(A)$ and then show this assertion by induction of the length of the subformulas of A, distinguishing the following cases:

(i) B is a literal. Then $\mathbb{K}_A(B)$ is the clause set $\{\{\overline{B}, Q_B\}, \{\neg Q_B, B\}\}$. Hence, since $\mathfrak{W}(Q_B) = \widehat{\mathfrak{V}}(B) = \widehat{\mathfrak{W}}(B)$, we clearly have $\widehat{\mathfrak{W}}[\mathbb{K}_A(B)] = \mathbf{t}$.

(ii) B is a formula $C \vee D$. Then $\mathbb{K}_A(B)$ is the clause set

$$\mathbb{K}_A(C) \cup \mathbb{K}_A(D) \cup \{\{\neg Q_C, Q_B\}, \{\neg Q_D, Q_B\}\{\neg Q_B, Q_C, Q_D\}\}.$$

The induction hypothesis gives us $\widehat{\mathfrak{W}}[\mathbb{K}_A(C)] = \mathbf{t}$ and $\widehat{\mathfrak{W}}[\mathbb{K}_A(D)] = \mathbf{t}$, hence

$$\widehat{\mathfrak{W}}[\mathbb{K}_A(C) \cup \mathbb{K}_A(D)] = \mathbf{t}. \tag{1}$$

In addition, the definition of $\mathfrak{W}$ implies

$$\mathfrak{W}(Q_B) = \widehat{\mathfrak{V}}(B), \tag{2}$$

$$\mathfrak{W}(Q_C) = \widehat{\mathfrak{V}}(C), \tag{3}$$

$$\mathfrak{W}(Q_D) = \widehat{\mathfrak{V}}(D). \tag{4}$$

From (2)–(4) we obtain by simple calculations that

$$\widehat{\mathfrak{W}}(\{\neg Q_C, Q_B\}) = \widehat{\mathfrak{W}}(\{\neg Q_D, Q_B\}) = \widehat{\mathfrak{W}}(\{\neg Q_B, Q_C, Q_D\}) = \mathbf{t}. \tag{5}$$

But now (1) and (5) imply that $\widehat{\mathfrak{W}}[\mathbb{K}_A(B)] = \mathbf{t}$, as required.

(iii) B is a formula $C \wedge D$. We simply proceed as in the previous case. □

Theorem 164 (Satisfiability equivalence)

1. *Let A be a formula in negation normal form without propositional constants and choose $(Q_B : B \in \mathrm{Sufo}(A))$ according to Definition 159. Then we have that*

$$A \text{ is satisfiable} \iff \mathbb{K}_A \cup \{\{Q_A\}\} \text{ is satisfiable.}$$

2. *If B is an arbitrary formula, we set* $\mathrm{Sat}(B) \coloneqq \mathbb{K}_{\sigma(B)} \cup \{\{Q_{\sigma(B)}\}\}$*, where σ is the operation introduced in Definition 154. Then we have*

$$B \text{ is satisfiable} \iff \mathrm{Sat}(B) \text{ is satisfiable.}$$

PROOF 1. To show the direction from left to right, we assume that A is satisfiable. Hence there is a valuation $\mathfrak{V}$ such that $\widehat{\mathfrak{V}}(A) = \mathbf{t}$. For the valuation $\mathfrak{W}$ defined as in Lemma 163 we then have $\mathfrak{W}(Q_A) = \mathbf{t}$. Moreover, this lemma also implies $\widehat{\mathfrak{W}}[\mathbb{K}_A] = \mathbf{t}$. Hence, $\widehat{\mathfrak{W}}[\mathbb{K}_A \cup \{\{Q_A\}\}] = \mathbf{t}$, implying that $\mathbb{K}_A \cup \{\{Q_A\}\}$ is satisfiable.

For the converse direction, let $\mathbb{K}_A \cup \{\{Q_A\}\}$ be satisfiable. Then there exists a valuation $\mathfrak{V}$ with $\widehat{\mathfrak{V}}[\mathbb{K}_A] = \mathbf{t}$ and $\mathfrak{V}(Q_A) = \mathbf{t}$. Now we apply Lemma 162 and conclude $\widehat{\mathfrak{V}}(A) = \mathbf{t}$. Hence A is satisfiable.

2. It follows from Definition 154 that B is satisfiable if and only if $\sigma(B)$ is satisfiable. Hence our second assertion is immediate from the first. □

Remark 165

1. For any formula A in negation normal form without propositional constants the clauses in clause set $\mathbb{K}_A$ contain at most three literals, and in view of Lemma 161 we know that the size of $\mathbb{K}_A$ is bounded by $3 \cdot \ell(A)$. Hence $\mathbb{K}_A \cup \{\{Q_A\}\}$ is linear in $\ell(A)$.

2. As we have seen earlier, given an arbitrary formula B, there is a polynomial transformation into the formula $\sigma(B)$ in negation normal form without propositional constants.

3. Altogether we thus know that for any formula B there exists a polynomial transformation into the equisatisfiable clause set $\mathrm{Sat}(B)$. This is in sharp contrast to the determination of the conjunctive normal form of B, which may be exponential in $\ell(B)$.

Once we have a clause set $\mathbb{K}$, the stage is set to test the satisfiability of $\mathbb{K}$ by resolution. Basically, all one has to do is to close $\mathbb{K}$ under forming resolvents (see below) and check whether the empty clause □ can be obtained in this way. If this is the case, $\mathbb{K}$ is not satisfiable, otherwise $\mathbb{K}$ is satisfiable.

Definition 166

1. *Given clauses $\Phi \cup \{P\}$ and $\Psi \cup \{\neg P\}$, the clause $\Phi \cup \Psi$ is called a* resolvent *of $\Phi \cup \{P\}$ and $\Psi \cup \{\neg P\}$.*

2. *The collection of all resolvents which can be formed from a clause set* $\mathbb{K}$ *is given by*

$$\text{Resolve}(\mathbb{K}) \coloneqq \{\Phi \cup \Psi : \textit{there exists a } P \textit{ such that } \Phi \cup \{P\}, \Psi \cup \{\neg P\} \in \mathbb{K}\}.$$

Example 167 Consider the two clauses $\{P, Q, R\}$ and $\{\neg P, \neg Q, R\}$. Then they have two different resolvents: $\{Q, R, \neg Q\}$ and $\{P, R, \neg P\}$.

Definition 168 *Let* $\mathbb{K}$ *be an arbitrary clause set.*

1. *For all natural numbers* n *we define the clause sets* $\mathfrak{R}_n(\mathbb{K})$ *by induction on* n *as follows:*

$$\mathfrak{R}_0(\mathbb{K}) \coloneqq \mathbb{K},$$

$$\mathfrak{R}_{n+1}(\mathbb{K}) \coloneqq \mathfrak{R}_n(\mathbb{K}) \cup \text{Resolve}(\mathfrak{R}_n(\mathbb{K})).$$

2. *Accordingly, the closure of* $\mathbb{K}$ *under resolution is the clause set*

$$\mathfrak{R}(\mathbb{K}) \coloneqq \bigcup_{n \in \mathbb{N}} \mathfrak{R}_n(\mathbb{K}).$$

Example 169 For the clause set $\mathbb{K} \coloneqq \{\{P_1, P_2, P_3\}, \{P_4\}, \{\neg P_5\}, \{\neg P_2, P_5, P_6\}\}$ we have:

$$\begin{aligned}
\mathfrak{R}_0(\mathbb{K}) &= \{\{P_1, P_2, P_3\}, \{P_4\}, \{\neg P_5\}, \{\neg P_2, P_5, P_6\}\},\\
\mathfrak{R}_1(\mathbb{K}) &= \{\{P_1, P_2, P_3\}, \{P_4\}, \{\neg P_5\}, \{\neg P_2, P_5, P_6\}, \{P_1, P_3, P_5, P_6\},\\
&\quad \{\neg P_2, P_6\}\},\\
\mathfrak{R}_2(\mathbb{K}) &= \{\{P_1, P_2, P_3\}, \{P_4\}, \{\neg P_5\}, \{\neg P_2, P_5, P_6\}, \{P_1, P_3, P_5, P_6\},\\
&\quad \{\neg P_2, P_6\}, \{P_1, P_3, P_6\}\}.
\end{aligned}$$

Since $\mathfrak{R}_2(\mathbb{K})$ is closed under resolution, we conclude $\mathfrak{R}_n(\mathbb{K}) = \mathfrak{R}_2(\mathbb{K})$ for all natural numbers $n \geq 2$, hence $\mathfrak{R}(\mathbb{K}) = \mathfrak{R}_2(\mathbb{K})$.

Before formulating and proving the correctness and completeness of resolution, we state a few properties of the clause sets $\mathfrak{R}_n(\mathbb{K})$ and $\mathfrak{R}(\mathbb{K})$.

Lemma 170 *Let* $\mathbb{K}$ *and* $\mathbb{L}$ *be arbitrary clause sets and* m *and* n *arbitrary natural numbers*

1. $\mathbb{K} \subseteq \mathbb{L} \implies \mathfrak{R}_n(\mathbb{K}) \subseteq \mathfrak{R}_n(\mathbb{L})$.
2. $m \leq n \implies \mathfrak{R}_m(\mathbb{K}) \subseteq \mathfrak{R}_n(\mathbb{K})$.

3. $\mathrm{Resolve}(\mathfrak{R}(\mathbb{K})) \subseteq \mathfrak{R}(\mathbb{K})$.

4. $\mathfrak{R}(\mathfrak{R}(\mathbb{K})) = \mathfrak{R}(\mathbb{K})$.

5. $\mathbb{K} \subseteq \mathfrak{R}(\mathbb{L}) \implies \mathfrak{R}(\mathbb{K}) \subseteq \mathfrak{R}(\mathbb{L})$.

6. If a clause Φ *belongs to* $\mathfrak{R}(\mathbb{K})$*, then there exists a finite subset* $\mathbb{K}_f$ *of* $\mathbb{K}$ *such that* $\Phi \in \mathfrak{R}(\mathbb{K}_f)$.

PROOF The first and the second assertion are obvious and obtained by trivial induction on n.

3. Take clauses $\Phi \cup \{P\}$ and $\Psi \cup \{\neg P\}$ from $\mathfrak{R}(\mathbb{K})$. Then there exist natural numbers m and n such that $\Phi \cup \{P\} \in \mathfrak{R}_m(\mathbb{K})$ and $\Psi \cup \{\neg P\} \in \mathfrak{R}_n(\mathbb{K})$. By the second assertion we therefore have $\Phi \cup \{P\} \in \mathfrak{R}_k(\mathbb{K})$ and $\Psi \cup \{\neg P\} \in \mathfrak{R}_k(\mathbb{K})$ for $k := \max(m, n)$. Therefore $\Phi \cup \Psi \in \mathfrak{R}_{k+1}(\mathbb{K}) \subseteq \mathfrak{R}(\mathbb{K})$.

4. From the third assertion we obtain $\mathfrak{R}_n(\mathfrak{R}(\mathbb{K})) \subseteq \mathfrak{R}(\mathbb{K})$ by straightforward induction on n. However, this implies $\mathfrak{R}(\mathfrak{R}(\mathbb{K})) = \mathfrak{R}(\mathbb{K})$.

5. If $\mathbb{K} \subseteq \mathfrak{R}(\mathbb{L})$, then $\mathfrak{R}(\mathbb{K}) \subseteq \mathfrak{R}(\mathfrak{R}(\mathbb{L})) = \mathfrak{R}(\mathbb{L})$ according to the first and fourth assertion.

6. For all natural numbers n and all clauses Φ we have

$$\Phi \in \mathfrak{R}_n(\mathbb{K}) \implies \text{there exists a finite } \mathbb{L} \subseteq \mathbb{K} \text{ such that } \Phi \in \mathfrak{R}_n(\mathbb{L}),$$

as can be easily seen by induction on n. Our assertion follows immediately. □

Lemma 171 *Let* $\mathbb{K}$ *be a clause set and* $\mathfrak{V}$ *a valuation which satisfies* $\mathbb{K}$*. Then we have for all natural numbers* n *and all clauses* Φ *that*

$$\Phi \in \mathfrak{R}_n(\mathbb{K}) \implies \widehat{\mathfrak{V}}(\Phi) = \mathbf{t}.$$

PROOF We proceed by complete induction on n. If $\Phi \in \mathfrak{R}_0(\mathbb{K})$, it is an element of $\mathbb{K}$, and $\widehat{\mathfrak{V}}(\Phi) = \mathbf{t}$ follows directly from the assumption that $\mathfrak{V}$ satisfies $\mathbb{K}$.

If $\Phi \in \mathfrak{R}_{n+1}(\mathbb{K})$, then $\Phi \in \mathfrak{R}_n(\mathbb{K})$ or $\Phi \in \mathrm{Resolve}(\mathfrak{R}_n(\mathbb{K}))$. In the first case we have $\widehat{\mathfrak{V}}(\Phi) = \mathbf{t}$ by the induction hypothesis. In the second case there exist clauses $\Upsilon \cup \{P\}$ and $\Psi \cup \{\neg P\}$ such that

$$\Phi = \Upsilon \cup \Psi, \tag{1}$$

$$\Upsilon \cup \{P\} \in \mathfrak{R}_n(\mathbb{K}), \tag{2}$$

$$\Psi \cup \{\neg P\} \in \mathfrak{R}_n(\mathbb{K}). \tag{3}$$

By the induction hypothesis we obtain from (2) and (3) that

$$\widehat{\mathfrak{V}}(\Upsilon \cup \{P\}) \;=\; \widehat{\mathfrak{V}}(\Psi \cup \{\neg P\}) \;=\; \mathbf{t}.$$

Therefore, $\widehat{\mathfrak{V}}(\Upsilon \cup \Psi) = \mathbf{t}$, and (1) immediately yields $\widehat{\mathfrak{V}}(\Phi) = \mathbf{t}$. □

Theorem 172 (Correctness of resolution)

For any satisfiable clause set $\mathbb{K}$, the clause set $\mathfrak{R}(\mathbb{K})$ does not contain the empty clause; i.e.

$$\mathbb{K} \textit{ is satisfiable} \quad\Longrightarrow\quad \square \notin \mathfrak{R}(\mathbb{K}).$$

PROOF Assume that $\mathbb{K}$ is satisfiable. Then there exists a valuation $\mathfrak{V}$ which satisfies $\mathbb{K}$. In view of the previous lemma we thus have $\widehat{\mathfrak{V}}(\Phi) = \mathbf{t}$ for all elements of $\mathfrak{R}(\mathbb{K})$. Since $\widehat{\mathfrak{V}}(\square) = \mathbf{f}$, cf. Remark 151, we conclude $\square \notin \mathfrak{R}(\mathbb{K})$. □

Theorem 173 (Completeness of resolution)

If for a clause set $\mathbb{K}$ the clause set $\mathfrak{R}(\mathbb{K})$ does not contain the empty clause, then $\mathbb{K}$ is satisfiable; i.e.

$$\square \notin \mathfrak{R}(\mathbb{K}) \quad\Longrightarrow\quad \mathbb{K} \textit{ is satisfiable.}$$

PROOF Suppose $\square \notin \mathfrak{R}(\mathbb{K})$. For every natural number n we define clauses Φ_n by induction on n:

$$\Phi_0 \;:=\; \square;$$

$$\Phi_{n+1} \;:=\; \begin{cases} \Phi_n \cup \{\mathfrak{p}_n\} & \text{if there exists no } \Psi \in \mathfrak{R}(\mathbb{K}) \text{ such that } \Psi \subseteq \Phi_n \cup \{\mathfrak{p}_n\}, \\ \Phi_n \cup \{\neg\mathfrak{p}_n\} & \text{otherwise.} \end{cases}$$

For all natural numbers n, these clause sets Φ_n have the following properties:

(1) $\Phi_n \subseteq \Phi_{n+1}$.

(2) For any $m \geq n$ we have $\mathfrak{p}_m \notin \Phi_n$ and $\neg\mathfrak{p}_m \notin \Phi_n$.

(3) For any $m < n$ we have $\mathfrak{p}_m \in \Phi_n$ or $\neg\mathfrak{p}_m \in \Phi_n$, but not $\{\mathfrak{p}_m, \neg\mathfrak{p}_m\} \subseteq \Phi_n$.

(4) There exists no $\Psi \in \mathfrak{R}(\mathbb{K})$ such that $\Psi \subseteq \Phi_n$.

Assertions (1)–(3) are clear; assertion (4) is now proved by induction on n.

$n = 0$. Since $\Phi_0 = \square$, assertion (4) follows in this case from $\square \notin \mathfrak{R}(\mathbb{K})$.

$n > 0$. Then $n = m + 1$ for some natural number m. Assume that (4) is not satisfied for this $m + 1$. According to the definition of Φ_{m+1} we then have $\Upsilon_1, \Upsilon_2 \in \mathfrak{R}(\mathbb{K})$ for which

$$\Upsilon_1 \subseteq \Phi_m \cup \{\mathfrak{p}_m\} \quad \text{and} \quad \Upsilon_2 \subseteq \Phi_m \cup \{\neg\mathfrak{p}_m\}.$$

From the induction hypothesis we deduce that, in addition,

$$\Upsilon_1 \nsubseteq \Phi_m \quad \text{and} \quad \Upsilon_2 \nsubseteq \Phi_m.$$

Hence there exist $\Upsilon_1' \subseteq \Phi_m$ and $\Upsilon_2' \subseteq \Phi_m$ with

$$\Upsilon_1 = \Upsilon_1' \cup \{\mathfrak{p}_m\} \quad \text{and} \quad \Upsilon_2 = \Upsilon_2' \cup \{\neg\mathfrak{p}_m\}.$$

But now, since $\Upsilon_1 \in \mathfrak{R}(\mathbb{K})$ and $\Upsilon_2 \in \mathfrak{R}(\mathbb{K})$, Lemma 170 yields $\Upsilon_1' \cup \Upsilon_2' \in \mathfrak{R}(\mathbb{K})$. Because of $\Upsilon_1' \cup \Upsilon_2' \subseteq \Phi_m$ this is a contradiction to the induction hypothesis. Hence our assumption is wrong, meaning that assertion (4) also is satisfied for $n > 0$.

Finally, we introduce a valuation $\mathfrak{V}$ by defining, for any $n \in \mathbb{N}$,

$$\mathfrak{V}(\mathfrak{p}_n) := \begin{cases} \mathbf{f} & \text{if } \mathfrak{p}_n \in \Phi_{n+1}, \\ \mathbf{t} & \text{if } \mathfrak{p}_n \notin \Phi_{n+1}. \end{cases}$$

Let Ψ be a clause with $\widehat{\mathfrak{V}}(\Psi) = \mathbf{f}$. We first determine the largest index n for which $\mathfrak{p}_n \in \Psi$ or $\neg\mathfrak{p}_n \in \Psi$ and conclude by (1) and the definition of $\mathfrak{V}$ that $\Psi \subseteq \Phi_{n+1}$. Hence we have shown:

(5) For any clause Ψ with $\widehat{\mathfrak{V}}(\Psi) = \mathbf{f}$ there exists a natural number n such that $\Psi \subseteq \Phi_{n+1}$.

To finish the proof of our completeness result, take a clause $\Psi \in \mathbb{K}$. Then $\Psi \in \mathfrak{R}(\mathbb{K})$, and (4) implies $\Psi \notin \Phi_n$ for all natural numbers n. Hence $\widehat{\mathfrak{V}}(\Psi) = \mathbf{t}$ by (5). So we have shown that $\mathfrak{V}$ satisfies the clause set $\mathbb{K}$. □

Corollary 174 *For any clause set* $\mathbb{K}$ *we have*

$$\square \notin \mathfrak{R}(\mathbb{K}) \iff \mathbb{K} \textit{ is satisfiable.}$$

If T is the finite theory $\{B_1, \ldots, B_n\}$ we can use resolution in order to find out whether A is a logical consequence of T: First we recall from the deduction theorem, see Theorem 112, that this is the case if and only if $B_1 \wedge \ldots \wedge B_n \to A$ is valid. Then we conclude from Lemma 100 that $B_1 \wedge \ldots \wedge B_n \to A$ is valid if and only if $\neg(B_1 \wedge \ldots \wedge B_n \to A)$ is not satisfiable. After transforming this formula into an equisatisfiable clause set, we can apply resolution to find out whether this is the case. Hence we have the following theorem.

Theorem 175 *Let T be the theory $\{B_1, \ldots, B_n\}$ and let A be an arbitrary formula. Then A is a logical consequence of T if and only if $\square \in \mathfrak{R}(\mathrm{Sat}(\neg(B_1 \wedge \ldots \wedge B_n \rightarrow A)))$, where $\mathrm{Sat}(\neg(B_1 \wedge \ldots \wedge B_n \rightarrow A))$ is the clause set introduced in Theorem 164; i.e.*

$$T \models A \quad \Longleftrightarrow \quad \square \in \mathfrak{R}(\mathrm{Sat}(\neg(B_1 \wedge \ldots \wedge B_n \rightarrow A))).$$

Resolution is an extremely powerful proof method for classical propositional Logic. The search space when using resolution is generally much smaller than, for example, when working with truth tables. Nevertheless, as proved in Haken [5], the worst case behavior of resolution is exponential. It has been shown there that in dealing with the pigeon formulas, exponentially many clauses are being generated.

2.11 Summary

This chapter dealt with classical propositional logic CP, the most basic logical system. After introducing the syntax and semantics of CP we turned to the question of finding out whether a given formula A is satisfiable or valid, and came up with a very simple algorithm which, however, is exponential in the number of atomic propositions occurring in A.

Theories are collections of formulas. They are often used in order to formally represent a given situation, and then the question arises whether a given assertion holds in this situation, i.e. whether A is a logical consequence of T. For finite theories $\{B_1, \ldots, B_n\}$ this question can be reduced, via the deduction theorem, to the question whether the formula $B_1 \wedge \ldots \wedge B_n \rightarrow A$ is valid.

Because of the significance of the satisfiability and validity problem, we spent some time on two important systems addressing this topic:

Proof search calculus PSC. This is a correct and complete system testing formulas in negation normal form on validity. It is goal driven in the sense that the syntactic structure of the formula A to be tested guides the build up of the so-called deduction chains which either form a proof (if A is valid) or provide a counter model (if A is not valid). Since any formula can be easily transformed into negation normal form, we thus can deal with all propositional formulas.

Resolution. This is probably the conceptually most simple deduction method for testing the satisfiability of clause sets. And since there is a polynomial transformation of any propositional formula into an equisatisfiable clause set, resolution is a reasonable method for testing the satisfiability of propositional formulas. It is not goal driven, but based on one type of rules only.

Bibliography

[1] W. Buchholz, *Diskrete Strukturen*, Vorlesungsskript Sommersemester 2009, Mathematisches Institut, Universität München.

[2] O. Deiser, *Einführung in die Mengenlehre*, 2nd ed., Springer, 2004.

[3] K. Devlin, *The Joy of Sets: Fundamentals of Contemporary Set Theory*, 2nd ed., Undergraduate Texts in Mathematics, Springer, 1993.

[4] R. Diestel, *Graphentheorie*, 4th ed., Springer-Lehrbuch Masterclass, Springer, 2010.

[5] A. Haken, *The intractability of resolution*, Theoretical Computer Science **39** (1985), no. 2-3, 297–308.

[6] P. R. Halmos, *Naive Set Theory*, 2nd ed., Undergraduate Texts in Mathematics, Springer, 1974.

[7] A. Leitsch, *The Resolution Calculus*, Springer, 1997.

[8] I. Niven, H.S. Zuckerman, and H.L. Montgoery, *An Introduction to the Theory of Numbers*, 5th ed., Wiley, 1991.

[9] R. Rivest, A. Shamir, and L. Adelman, *A method for obtaining digital signatures and public-key cryptosystems*, Communications of the ACM **21** (1978), no. 2, 120–126.

[10] J.A. Robinson, *A machine-oriented logic based on the resolution principle*, Journal of the ACM **12** (1965), no. 1, 23–41.

[11] K. Schütte, *Proof Theory*, Grundlehren der Mathematischen Wissenschaften, Springer, 1977.

[12] H. Schwichtenberg, *Diskrete Strukturen*, Vorlesungsskript Sommersemester 2008, Mathematisches Institut, Universität München.