Discrete Mathematics

Hans Cuypers

October 11, 2007

Contents

1. Relations **4**
1.1. Binary relations . . . 4
1.2. Equivalence relations . . . 6
1.3. Relations and Directed Graphs . . . 8
1.4. Composition of Relations . . . 9
1.5. Transitive Closure . . . 10
1.6. Exercises . . . 14

2. Maps **16**
2.1. Maps . . . 16
2.2. Special Maps . . . 18
2.3. Exercises . . . 21

3. Order relations **22**
3.1. Posets . . . 22
3.2. Maximal and Minimal Elements . . . 24
3.3. Exercises . . . 26

4. Recursion and Induction **28**
4.1. Recursion . . . 28
4.2. Linear recurrences . . . 32
4.3. Natural Induction . . . 34
4.4. Strong Induction and Minimal Counter Examples . . . 36
4.5. Structural Induction . . . 38
4.6. Exercises . . . 39

5. Bounding some recurrences **42**
5.1. Growth of a function . . . 42
5.2. The Master Theorem . . . 44
5.3. Exercises . . . 46

6. Graphs **47**
6.1. Graphs . . . 47
6.2. Some special graphs . . . 47
6.3. Euler and Hamilton Cycles . . . 48
6.4. Spanning Trees and Search Algorithms . . . 49
6.5. Networks . . . 51
6.6. Planar Graphs . . . 54
6.7. Graph Colorings . . . 58
6.8. Exercises . . . 58

7. Lattices and Boolean Algebras **61**
7.1. Lattices 61
7.2. Boolean Algebras 64
7.3. Examples from Logic and Electrical Engineering 67
7.4. The Structure of Boolean Algebras 68
7.5. Exercises 71

1. Relations

1.1. Binary relations

A (binary) *relation* R between the sets S and T is a subset of the cartesian product $S \times T$.

If $(a, b) \in R$, we say a is in relation R to be b. We denote this by aRb. The set S is called the *domain* of the relation and the set T the *codomain*. If $S = T$ we say R is a relation on S.

Example 1.1.1 We give some examples:

1. "Is the mother of" is a relation between the set of all females and the set of all people. It consists of all the pairs (person 1, person 2 where person 1 is the mother of person 2.
2. "There is a train connection between" is a relation between the cities of the Netherlands.
3. The identity relation "=" is a relation on a set S. This relation is often denoted by I. So,
$$I = \{(s, s) \mid s \in S\}.$$
4. We say an integer n divides an integer m, notation $n \mid m$, if there is an element $q \in \mathbb{Z}$ such that $an = m$. Divides $\mid$ is a relation on $\mathbb{Z}$ consisting of all the pairs $(n, m) \in \mathbb{Z} \times \mathbb{Z}$ with $n \mid m$.
5. "Greater than" $>$ or "less than" $<$ are relations on $\mathbb{R}$.
6. $R = \{(0, 0), (1, 0), (2, 1)\}$ is a relation between the sets $S = \{0, 1, 2\}$ and $T = \{0, 1\}$.
7. $R = \{(x, y) \in \mathbb{R}^2 \mid y = x^2\}$ is a relation on $\mathbb{R}$.
8. Let Ω be a set, then "is a subset of" $\subseteq$ is a relation on the set S of all subsets of Ω.

Besides binary relations one can also consider n-ary relations with $n \geq 0$. An n-ary relation R on the sets $S_1, \ldots, S_n$ is a subset of the cartesian product $S_1 \times \cdots \times S_n$. In these notes we will restrict our attention to binary relations. Unless stated otherwise, a relation will be assumed to be binary.

Let R be a relation from a set S to a set T. Then for each element $a \in S$ we define $[a]_R$ to be the set
$$[a]_R := \{b \in S \mid aRb\}.$$

(Sometimes this set is also denoted by $R(a)$.) This set is called the (R-) image of a. For $b \in T$ the set

$$_R[b] := \{a \in S \mid aRb\}$$

is called the (R-) pre-image of b or R-fiber of b.

Definition 1.1.2 If $S = \{s_1, \ldots, s_n\}$ and $T = \{t_1, \ldots, t_m\}$ are finite sets and $R \subseteq S \times T$ is a binary relations, then the *adjacency matrix* A_R of the relation R is the $n \times m$ matrix whose rows are indexed by S and columns by T defined by

$$\begin{aligned} A_{s,t} &= 1 \text{ if } (s,t) \in R; \\ &= 0 \text{ otherwise.} \end{aligned}$$

Notice that a presentation of the adjacency matrix of a relation is defined upto permutations of the rows and columns of the matrix. If the sets S and T are equal, then it is customary to put the rows in the same order as the columns.

If $s \in S$, then $[s]_R$ consists of those $t \in T$ such that the entry t of the row s in A_R equals 1. For $t \in T$ the set $_R[t]$ consists of the nonzero entries in the column of t.

Example 1.1.3

1. The adjacency matrix of the relation $R = \{(0,0),(1,0),(2,1)\}$ between the sets $S = \{0,1,2\}$ and $B = \{0,1\}$ equals

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

   (We number rows from top to bottom and columns from left to right.)

2. The adjacency matrix of the identity relation on a set S of size n is the $n \times n$ identity matrix

$$I_n = \begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & \cdots & 0 & 1 \end{pmatrix}.$$

3. The adjacency matrix of relation $\leq$ on the set $\{1,2,3,4,5\}$ is the upper triangular matrix

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

Some relations have special properties:

Definition 1.1.4 Let R be a relation on a set S. Then R is called

- Reflexive if for all $x \in S$ we have $(x, x) \in R$;
- Irreflexive if for all $x \in S$ we have $(x, x) \notin R$;
- Symmetric if for all $x, y \in S$ we have xRy implies yRx;
- Antisymmetric if for all $x, y \in S$ we have that xRy and yRx implies $x = y$;
- Transitive if for all $x, y, z \in S$ we have that xRy and yRz implies xRz.

Example 1.1.5 We consider some of the examples given above:

1. "Is the mother of" is a relation on the set of all people. This relation is irreflexive, antisymmetric and not transitive.
2. "There is a train connection between" is a symmetric and transitive relation.
3. "=" is a reflexive, symmetric and transitive relation on a set S.
4. divides $|$ is a reflexive, antisymmetric and transitive relation on $\mathbb{N}$.
5. "Greater than" $>$ or "less than" $<$ on $\mathbb{R}$ are irreflexive, antisymmetric and transitive.
6. The relation $R = \{(x, y) \in \mathbb{R}^2 \mid y = x^2\}$ is not reflexive nor irreflexive.

If R is a relation on a finite set S, then special properties like reflexivity, symmetry and transitivity can be read of from the adjacency matrix A. For example, the relation R on a set S is reflexive if and only if the main diagonal of A only contains 1's, i.e., $A_{s,s} = 1$ for all $s \in S$.

The relation R is symmetric if and only if the transposed matrix $A^\top$ of A equals A. (The *transposed matrix* $M^\top$ of an $n \times m$ matrix M is the $m \times n$ matrix with entry i, j equal to $M_{j,i}$.)

1.2. Equivalence relations

As we noticed in the above example, "being equal" is a reflexive, symmetric and transitive relation on any set S. Relations having these three properties deserve some special attention.

Definition 1.2.1 A relation R on a set S is called an *equivalence relation* on S if and only if it is reflexive, symmetric and transitive.

Example 1.2.2 Consider the plane $\mathbb{R}^2$ and in it the set S of straight lines. We call two lines parallel in S if and only if they are equal or do not intersect. Notice that two lines in S are parallel if and only if their slope is equal. Being parallel defines an equivalence relation on the set S.

Example 1.2.3 Fix $n \in \mathbb{Z}$, $n \neq 0$, and consider the relation R on $\mathbb{Z}$ by aRb if and only if $a - b$ is divisible by n. We also write $a = b \mod n$. As we will see in Chapter 2 of [1], this is indeed an equivalence relation.

Example 1.2.4 Let Π be a partition of the set S, i.e., Π is a set of nonempty subsets of S such that each element of S is in a unique member of Π. In particular, the union of all members of Π yields the whole set S and any two members of Π have empty intersection.

We define the relation R_Π as follows: $a, b \in S$ are in relation R_Π if and only if there is a subset X of S in Π containing both a and b. we check that the relation R_Π is an equivalence relation on S.

- Reflexivity. Let $a \in S$. Then there is an $X \in \Pi$ containing a. Hence $a, a \in X$ and $aR_\Pi a$
- Symmetry. Let $aR_\Pi b$. then there is an $X \in \Pi$ with $a, b \in X$. But then also $b, a \in X$ and $bR_\Pi a$.
- Transitivity. If $a, b, c \in S$ with $aR_\Pi b$ and $bR_\Pi c$, then there are $X, Y \in \Pi$ with $a, b \in X$ and $b, c \in Y$. However, then b is in both X and Y. But then, as Π partitions S, we have $X = Y$. So $a, c \in X$ and $aR_\Pi c$.

The following theorem implies that every equivalence relation on a set S can be obtained as a partition of the set S.

Lemma 1.2.5 *Let R be an equivalence relation on a set S. If $b \in [a]_R$, then $[b]_R = [a]_R$.*

Proof. Suppose $b \in [a]_R$. Thus aRb. If $c \in [b]_R$, then bRc and, as aRb, we have by transitivity aRc. In particular, $[b]_R \subseteq [a]_R$.

Since, by symmetry of R, aRb implies bRa and hence $a \in [b]_R$, we similarly get $[a]_R \subseteq [b]_R$. □

Theorem 1.2.6 *Let R be an equivalence relation on a set S. Then the set of R-equivalence classes partitions the set S.*

Proof. Let Π_R be the set of R-equivalence classes. Then by reflexivity of R we find that each element $a \in S$ is inside the class $[a]_R$ of Π_R.

If an element $a \in S$ is in the classes $[b]_R$ and $[c]_R$ of Π, then by the previous lemma we find $[b]_R = [a]_R$ and $[b]_R = [c]_R$. In particular $[b]_R$ equals $[c]_R$. Thus each element $a \in S$ is inside a unique member of Π_R, which therefore is a partition of S. □

1.3. Relations and Directed Graphs

A *directed edge* of a set V is an element of $V \times V$. If $e = (v, w)$ then v is called its *tail* and w its *head*. Both v and w are called *endpoints* of the edge e. The *reverse* of the edge e is the edge (w, v).

A *directed graph* (also called *digraph*) $\Gamma = (V, E)$ consists of a set V of *vertices* and a subset E of $V \times V$ of (directed) *edges*. The elements of V are called the *vertices* of Γ and the elements of E the edges of Γ. Clearly, the edge set of a directed graph is a relation on the set of vertices. Conversely, if R is a binary relation on a set S, then R defines a *directed graph* (S, R) (also called *digraph*) on the set S, which we denote by Γ_R. Hence there is a one-to-one correspondence between directed graphs and relations. It is often convenient to switch from a relation to the corresponding digraph or back.

In this subsection we introduce some graph theoretical language and notation to be used in the sequel.

Suppose $\Gamma = (V, E)$ is a digraph. A *path* from v to w, where $v, w \in V$, is a sequence $v_0, v_1, \ldots, v_k$ of vertices with $v_0 = v$, $v_k = w$ and $(v_i, v_{i+1}) \in E$ for all $0 \leq i < k$. The *length* of the path is k. A path is called *simple* if all the vertices v_0 up to v_{k-1} are distinct. A *cycle* is a path form v to v and is called simple if the path is simple.

If $v, w \in V$ are vertices of the digraph Γ, then the *distance* from v to w is the minimum of the lengths of the paths from v to w. (The distance is set to ∞ if there is no path from v to w.)

The digraph is called *connected* if for any two vertices v and w there is a path from v to w or from w to v. It is called *strongly connected* if there exist paths in both directions.

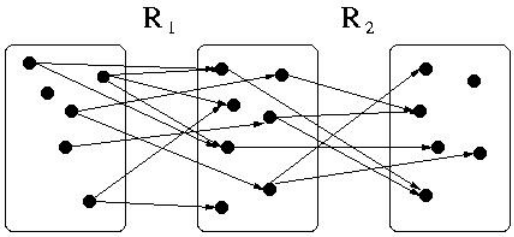
If W is a subset of V, then the induced subgraph of Γ on W is the digraph $(W, E \cap W \times W)$. A *connected component* C of Γ is a maximal subset of V such that the induced subgraph is connected. This means that the induced subgraph is connected and there is no path between a vertex inside and one outside C.

A *strongly connected component* is a maximal subset of V such that the induced subgraph is strongly connected. This means that the induced subgraph is strongly connected and there are no vertices $v \in C$ and $w \notin C$ such that there are paths in both directions between v and w.

1.4. Composition of Relations

If R_1 and R_2 are two relations between a set S and a set T, then we can form new relations between S and T by taking the intersection $R_1 \cap R_2$ or the union $R_1 \cup R_2$. Also the complement of R_2 in R_1, $R_1 - R_2$, is a new relation. Furthermore we can consider a relation $R^\top$ (sometimes also denoted by R^{-1}, $R^\sim$ or $R^\vee$) from T to S as the relation $\{(t,s) \in T \times S \mid (s,t) \in R\}$.

Another way of making new relations out of old ones is the following. If R_1 is a relation between S and T and R_2 is a relation between T and U then the *composition or product* $R = R_1; R_2$ (sometimes denoted by $R_2 \circ R_1$ or $R_1 * R_2$) is the relation between S and U defined by sRu if and only if there is a $t \in T$ with sR_1t and tR_2u.



Example 1.4.1 Suppose R_1 is the relation $\{(1,2),(2,3),(3,3),(2,4)\}$ from $\{1,2,3\}$ to $\{1,2,3,4\}$ and R_2 the relation $\{(1,a),(2,b),(3,c),(4,d)\}$ from $\{1,2,3,4\}$ to $\{a,b,c,d\}$. Then $R_1; R_2$ is the relation $\{(1,b),(2,c),(3,c),(2,d)\}$ form $\{1,2,3\}$ to $\{a,b,c,d\}$.

Suppose R_1 is a relation from S to T and R_2 a relation from T to U with adjacency matrices A_1 and A_2, repsectively. Consider the matrix product $M = A_1A_2$. An entry $M_{s,u}$ is obtained by multiplying row s from A_1 with column u from A_2 and equals the number of $t \in T$ with $(s,t) \in R_1$ and $(t,u) \in R_2$.

Notice, if $R_1 = R_2$, then entry s,t equals the number of paths of length 2 in Γ_R starting in s and ending in t.

The adjacency matrix A of $R_1; R_2$ can be obtained from M by replcing every nonzero entry by a 1.

Example 1.4.2 Suppose R_1 $\{(1,2),(2,3),(3,3),(2,4),(3,1)\}$ from $\{1,2,3\}$ to $\{1,2,3,4\}$ and R_2 the relation $\{(1,1),(2,3),(3,1),(3,3),(4,2)\}$ from $\{1,2,3,4\}$ to $\{1,2,3\}$. Then the adjacency matrices A_1 and A_2 for R_1 and R_2 are

$$A_1 = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}, \; A_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}.$$

The product of these matrices equals

$$M = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \\ 2 & 0 & 1 \end{pmatrix}.$$

So, the adjacency matrix of $R_1; R_2$ is

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

Proposition 1.4.3 *Suppose R_1 is a relation from S to T, R_2 a relation from T to U and R_3 a relation from U to V. Then $R_1; (R_2; R_3) = (R_1; R_2); R_3$.*

Proof. Suppose $s \in S$ and $v \in V$ with $sR_1; (R_2; R_3)v$. Then we can find a $t \in T$ with sR_1t and $t(R_2; R_3)v$. But then there is also a $u \in U$ with tR_2u and uR_3v. For this u we have $sR_1; R_2u$ and uR_3v and hence $s(R_1; R_2); R_3v$.

Similarly, if $s \in S$ and $v \in V$ with $s(R_1; R_2); R_3v$, then we can find a $u \in U$ with $s(R_1; R_2)u$ and uR_3v. But then there is also a $t \in T$ with sR_1t and tR_2u. For this t we have $tR_2; R_3u$ and sR_1t and hence $sR_1; (R_2; R_3)v$. □

Let R be a relation on a set S and denote by I the identity relation on S, i.e., $I = \{(a, b) \in S \times S \mid a = b\}$. Then we easily check that $I; R = R; I = R$.

Let R be a relation on a set S and consider the directed graph Γ_R with vertex set S and edge set R. Then two vertices a and b are in relation $R^2 = R; R$, if and only if there is a $c \in S$ such that both (a, c) and $(c, b) \in R$. Thus aR^2b if and only if there is a path of length 2 from a to b.

For $n \in \mathbb{N}$, the n-th power R^n of the relation R is recursively defined by $R^0 = I$ and $R^{n+1} = R; R^n$. Two vertices a and b are in relation R^n if and only if, inside Γ_R, there is a path from a to b of length n.

We notice that whenever R is reflexive, we have $R \subseteq R^2$ and thus also $R \subseteq R^n$ for all $n \in \mathbb{N}$ with $n \geq 1$. Actually, a and b are then in relation R^n if and only if they are at distance $\leq n$ in the graph Γ_R.

1.5. Transitive Closure

Lemma 1.5.1 *Let $\mathcal{C}$ be a collection of relations R on a set S. If all relations R in $\mathcal{C}$ are transitive (symmetric or reflexive), then the relation $\bigcap_{R \in \mathcal{C}} R$ is also transitive (symmetric or reflexive, respectively).*

Proof. Let $\overline{R} = \bigcap_{R \in \mathcal{C}} R$. Suppose all members of $\mathcal{C}$ are transitive. Then for all $a, b, c \in S$ with $a\overline{R}b$ and $b\overline{R}c$ we have aRb and bRc for all $R \in \mathcal{C}$. Thus by transitivity

of each $R \in \mathcal{C}$ we also have aRc for each $R \in \mathcal{C}$. Thus we find $a\overline{R}c$. Hence $\overline{R}$ is also transitive.

The proof for symmetric or reflexive relations is left to the reader. □

The above lemma makes it possible to define the *reflexive, symmetric or transitive closure* of a relation R on a set S. It is the the smallest reflexive, symmetric or transitive relation containing R. This means, as follows from Lemma 1.5.1, it is the intersection $\bigcap_{R' \in \mathcal{C}} R'$, where $\mathcal{C}$ is the collection of all reflexive, symmetric or transitive relations containing R. Indeed, the above lemma implies that $\bigcap_{R' \in \mathcal{C}} R'$ is the smallest transitive (symmetric or reflexive) relation containing R if we take for $\mathcal{C}$ the appropriate set of all transitive (symmetric or reflexive) relations containing R.

Example 1.5.2 Suppose

$$R = \{(1,2),(2,2),(2,3),(5,4)\}$$

is a relation on $S = \{1,2,3,4,5\}$.

The reflexive closure of R is then the relation

$$\{(1,1),(1,2),(2,2),(2,3),(3,3),(4,4),(5,5),(5,4)\}.$$

The symmetric closure equals

$$\{(1,2),(2,1),(2,2),(2,3),(3,2),(5,4),(4,5)\}.$$

And, finally, the transitive closure of R equals

$$\{(1,2),(2,2),(2,3),(1,3),(5,4)\}.$$

One easily checks that the reflexive closure of a relation R equals the relation $I \cup R$ and the symmetric closure equals $R \cup R^{\top}$. The transitive closure is a bit more complicated. It contains $R, R^2, \ldots$. In particular, it contains $\bigcup_{n>0} R^n$, and, as we will show below, is equal to it.

Proposition 1.5.3 *$\bigcup_{n>0} R^n$ is the transitive closure of the relation R.*

Proof. Define $\bar{R} = \bigcup_{n>0} R^n$. We prove transitivity of $\bar{R}$. Let $a\bar{R}b$ and $b\bar{R}c$, then there are sequences $a_1 = a, \ldots, a_k = b$ and $b_1 = b, \ldots, b_l = c$ with $a_i R a_{i+1}$ and $b_i R b_{i+1}$. But then the sequence $c_1 = a_1 = a, \ldots, c_k = a_k = b_1, \ldots, c_{k+l-1} = b_l = c$ is a sequence from a to c with $c_i R c_{i+1}$. Hence $aR^{k+l-2}c$ and $a\bar{R}c$. □

The transitive, symmetric and reflexive closure of a relation R is an equivalence relations. In terms of the graph Γ_R, the equivalence classes are the strongly connected components of Γ_R.

Example 1.5.4 If we consider the whole World Wide Web as a set of documents, then we may consider two documents to be in a (symmetric) relation R if there is a hyperlink from one document to the another.

The reflexive and transitive closure of the relation R defines a partition of the web into independent subwebs.

Example 1.5.5 Let S be the set of railway stations in the Netherlands. Two stations a and b are in relation R if there is a train running directly from a to b.

If $\bar{R}$ denotes the transitive closure of R, then the railway stations in $[a]_{\bar{R}}$ are exactly those stations you can reach by train when starting in a.

Algorithm 1.5.6 [Warshall's Algorithm for the Transitive closure] Suppose a relation R on a finite set S of size n is given by its adjacency matrix A_R. Then Warshall's Algorithm is an efficient method for finding the adjacency matrix of the transitive closure of the relation R.

In the algorithm we construct a sequence $W^0, \ldots W^n$ of $n \times n$-matrices with only 0's and 1's, starting with $W^0 = A_R$ and ending with the adjacency matrix of the transitive closure of R.

On the set of rows of a matrix we define the operation $\vee$ by $(s_1, \ldots, s_n) \vee (t_1, \ldots, t_n) = (\max(t_1, s_1), \ldots, \max(t_n, s_n))$.

Now the algorithm proceeds as follows.

- The initial step is to set W^0 equal to A_R.
- For $k \geq 1$, the matrix W^k equals W^{k-1} if the k^{th} row r is completely zero and otherwise is obtained from W^{k-1} by replacing each nonzero row s having a 1 at position k by $r \vee s$.
- The matrix W^n is given as the adjacency matrix of the transitive closure of R.

To prove that the resulting matrix W^n is indeed the adjacency matrix of the transitive closure of R, we argue as follows. We claim that the following holds:

Claim. The matrix W^k has a 1 at entry i, j if and only if there is a path

$$a_i = v_0, v_1, \ldots, v_{l-1}, v_l = a_j$$

from a_i to a_j with $v_1, \ldots, v_{l-1}$ in $\{a_1, \ldots, a_k\}$ in the graph Γ_R.

Indeed, for $k = 0$ the set $\{a_1, \ldots, a_k\}$ is empty, so the claim true. In W^1 we only have a one at entry i, j if and only if (a_i, a_j) was already in R, or (a_i, a_1) and (a_1, a_j) are in R and there is a path a_i, a_1, a_j from a_i to a_j only using vertices in $\{a_1\}$.

Now suppose for some $0 \leq l \leq n-1$ an entry i, j in W^l is 1 if and only if there is a path from a_i to a_j only using vertices from $\{a_1, \ldots, a_k\}$. Then a new 1 at entry

i, j in W^{l+1} is only added to W^l if there are paths from a_i to a_{l+1} and from a_{l+1} to a_j using only vertices in $\{a_1, \ldots, a_l\}$. Hence entry i, j in W^{l+1} is only 1 if there is a path from a_i to a_j only using vertices from $\{a_1, \ldots, a_{l+1}\}$.

Since for $l = 1$, our claim is true, the above shows that it is also true for $l = 2$. But then also for $l = 3$ and so on. Thus our claim holds for all k and in particular for $k = n$. But that means that entry i, j in W^n is equal to 1 if and only if there is a path from a_i to a_j. So W_n is indeed the adjacency matrix of the transitive closure of R.

The above proof is an example of a *proof by induction*. Later, in Chapter 4, we will encounter more examples of proofs by induction.

Example 1.5.7 Consider the relation R on $\{1, 2, 3, 4, 5\}$ given by the adjacency matrix

$$A = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

We will run Warshall's algorithm to find the adjacency matrix of the transitive closure of R.

We start with $W^0 = A$. As only row 1 and 3 have a 1 at position 1, we get

$$W^1 = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

Now row 1, 2 and 3 have a 1 at position 2 and we find

$$W^2 = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

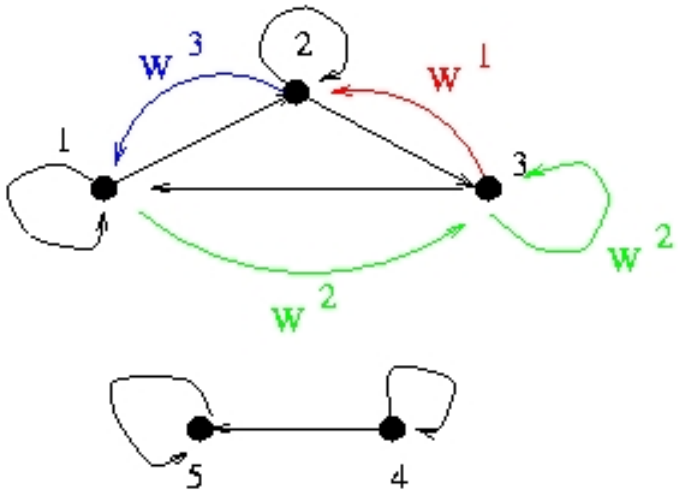
The first three rows do have now a 1 at position 3, so W^3 equals

$$W^3 = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

As only row 4 has a 1 at position 4, we also have $W^4 = W^3$. Finally W^5 also equals

$$W^5 = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

In the corresponding graph we see that edges are added in the following way:



1.6. Exercises

Exercise 1.6.1 Which of the following relations on the set $S = \{1, 2, 3, 4\}$ is reflexive, irreflexive, symmetric, antisymmetric or transitive?

1. $\{(1, 3), (2, 4), (3, 1), (4, 2)\}$;
2. $\{(1, 3), (2, 4)\}$;
3. $\{(1, 1), (2, 2), (3, 3), (4, 4), (1, 3), (2, 4), (3, 1), (4, 2)\}$;
4. $\{(1, 1), (2, 2), (3, 3), (4, 4)\}$;
5. $\{(1, 1), (2, 2), (3, 3), (4, 4), (1, 2), (2, 3), (3, 4), (4, 3), (3, 2), (2, 1)\}$.

Exercise 1.6.2 Let $A = \{1, 2, 3, 4\}$ and $R_1 = \{(1, 2), (1, 3), (2, 4), (2, 2), (3, 4), (4, 3)\}$ and $R_2 = \{(1, 1), (1, 2), (3, 1), (4, 3), (4, 4)\}$. Compute $R_1; R_2$ and $R_2; R_1$. Is the composition of relations commutative?

Exercise 1.6.3 Compute for each of the relations R in Exercise 1.6.1 the adjacency matrix and draw the digraph Γ_R.

Exercise 1.6.4 Compute for each of the relations R in Exercise 1.6.1 the adjacency matrix of R^2.

Exercise 1.6.5 Compute for each of the relations in Exercise 1.6.1 the reflexive closure, the symmetric closure and the transitive closure.

Exercise 1.6.6 Suppose R is a reflexive and transitive relation on S. Show that $R^2 = R$.

Exercise 1.6.7 Suppose R_1 and R_2 are two relations from the finite set S to the finite set T with adjacency matrices A_1 and A_2, respectively.

What is the adjacency matrix of the relation $R_1 \cap R_2$, $R_1 \cup R_2$, or $R_1^\top$?

Exercise 1.6.8 Suppose R_1 and R_2 are two relations on a set S. Let R be the product $R_1; R_2$. Prove or disprove the following statements

1. If R_1 and R_2 are reflexive, then so is R.
2. If R_1 and R_2 are irreflexive, then so is R.
3. If R_1 and R_2 are symmetric, then so is R.
4. If R_1 and R_2 are antisymmetric, then so is R.
5. If R_1 and R_2 are transitive, then so is R.

Exercise 1.6.9 Use Warshall's algorithm to compute the transitive closure of the relations given by the adjacency matrices

$$\begin{pmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \end{pmatrix} \text{ and } \begin{pmatrix} 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Exercise 1.6.10 Implement Warshall's algorithm in your favorite programming language.

2. Maps

2.1. Maps

Examples of maps are the well known functions $f : \mathbb{R} \to \mathbb{R}$ given by $f(x) = x^2$, $f(x) = \sin x$, or $f(x) = \frac{1}{x^2+1}$. We can view these maps as relations on $\mathbb{R}$. Indeed, the function $f : \mathbb{R} \to \mathbb{R}$ can be viewed as the relation $\{(x, y) \mid y = f(x)\}$. Actually, maps are special relations:

Definition 2.1.1 A relation F from a set A to a set B is called a *map* or *function* from A to B if for each $a \in A$ there is one and only one $b \in B$ with aFb.

If F is a map from A to B, we write this as $F : A \to B$. Moreover, if $a \in A$ and $b \in B$ is the unique element with aFb, then we write $b = F(a)$.

A *partial map* F from a set A to a set B is a relation with the property that for each $a \in A$ there is at most one b with aFb. In other words, it is a map from a subset A' of A to B.

Example 2.1.2 We have encountered numerous examples of maps. Below you will find some familiar ones.

1. polynomial functions like $f : \mathbb{R} \to \mathbb{R}$, with $f(x) = x^3$ for all x.
2. goneometric functions like cos, sin and tan.
3. $\sqrt{} : \mathbb{R}^+ \to \mathbb{R}$, taking square roots.
4. $\ln : \mathbb{R}^+ \to \mathbb{R}$, the natural logarithm.

If $f : A \to B$ and $g : B \to C$, then we can consider the product $f; g$ as a relation from A to C. We also use the notation $g \circ f$ and call it the composition of f and g. We prefer the latter notation for the composition of functions, as for all $a \in A$ we have

$$(g \circ f)(a) = g(f(a)).$$

Proposition 2.1.3 *Let $f : A \to B$ and $g : B \to C$ be maps, then the composition $g \circ f$ is a map from A to C.*

Proof. Let $a \in A$, then $g(f(a))$ is an element in C in relation $f; g$ with a. If $c \in C$ is an element in C that is in relation $f; g$ with a, then there is a $b \in B$ with afb and bgc. But then, as f is a map, $b = f(a)$ and, as g is a map, $c = g(b)$. Hence $c = g(b) = g(f(a))$. □

Let A and B be two sets and $f : A \to B$ a map from A to B. The set A is called the *domain* of f, the set B the *codomain*. If $a \in A$, then the element $b = f(a)$ is called

the *image* of a under f. The subset of B consisting of the images of the elements of A under f is called the *image* or *range* of f and is denote by $\mathrm{Im}(f)$. So

$$\mathrm{Im}(f) = \{b \in B \mid \text{there is a } a \in A \text{ with } b = f(a)\}.$$

If A' is a subset of A, then the image of A' under f is the set $f(A') = \{f(a) \mid a \in A'\}$. So, $\mathrm{Im}(f) = f(A)$.

If $a \in A$ and $b = f(a)$, then the element a is called a pre-image of b. Notice that b can have more than one pre-image. Indeed if $f : \mathbb{R} \to \mathbb{R}$ is given by $f(x) = x^2$ for all $x \in \mathbb{R}$, then both -2 and 2 are pre-images of 4. The set of all pre-images of b is denoted by $f^{-1}(b)$. So,

$$f^{-1}(b) = \{a \in A \mid f(a) = b\}.$$

If B' is a subset of B then the pre-image of B', denoted by $f^{-1}(B')$ is the set of elements a from A that are mapped to an element b of B'. In particular,

$$f^{-1}(B') = \{a \in A \mid f(a) \in B'\}.$$

Example 2.1.4

1. Let $f : \mathbb{R} \to \mathbb{R}$ with $f(x) = x^2$ for all $x \in \mathbb{R}$. Then $f^{-1}([0, 4]) = [-2, 2]$.
2. Consider the map $\bmod 8$ from $\mathbb{Z}$ to $\mathbb{Z}$. The inverse image of 3 is the set $\{\ldots, -5, 3, 11, \ldots\}$.

Theorem 2.1.5 *Let $f : A \to B$ be a map.*

- *If $A' \subseteq A$, then $f^{-1}(f(A')) \supseteq A'$.*
- *If $B' \subseteq B$, then $f(f^{-1}(B')) \subseteq B'$.*

Proof. Let $a' \in A'$, then $f(a') \in f(A')$ and hence $a' \in f^{-1}(f(A'))$. Thus $A' \subseteq f^{-1}(f(A'))$.

Let $a \in f^{-1}(B')$, then $f(a) \in B'$. Thus $f(f^{-1}(B')) \subseteq B'$. □

Example 2.1.6 Let $f : \mathbb{R} \to \mathbb{R}$ be defined by $f(x) = x^2$ for all $x \in \mathbb{R}$. Then $f^{-1}(f([0, 1]\}))$ equals $[-1, 1]$ and thus properly contains $[0, 1]$. Moreover, $f(f^{-1}([-4, 4])) = [0, 4]$ which is properly contained in $[-4, 4]$. This shows that we can have strict inclusions in the above theorem.

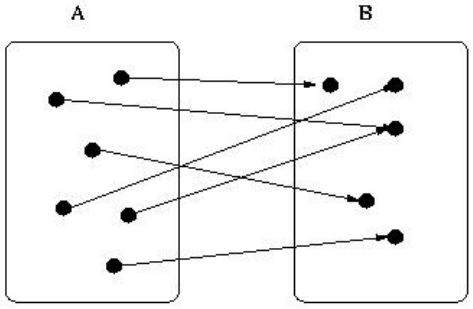
Theorem 2.1.7 *Let $f : A \to B$ and $g : B \to C$ be maps. Then* $\mathrm{Im}(g \circ f) = g(f(A)) \subseteq \mathrm{Im}(g)$.

2.2. Special Maps

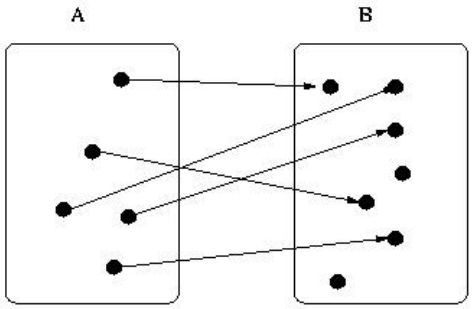
Definition 2.2.1 A map $f : A \to B$ is called *surjective*, if for every $b \in B$ there is an $a \in A$ with $b = f(a)$. In other words if $\mathrm{Im}(f) = B$.

The map f is called *injective* if for each $b \in B$, there is at most one a with $f(a) = b$. So the pre-image of b is either empty or consist of a unique element. In other words, f is injective if for any elements a and a' from A we find that $f(a) = f(a')$ implies $a = a'$.

The map f is *bijective* if it is both injective and surjective. So, if for each $b \in B$ there is a unique $a \in A$ with $f(a) = b$.

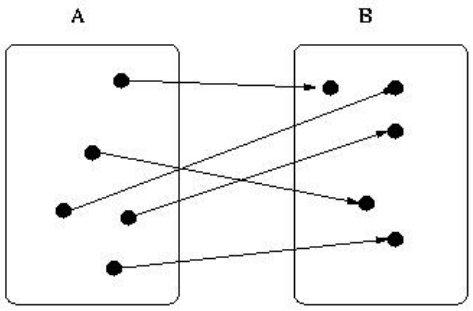


A surjective map from A to B



An injective map from A to B

Example 2.2.2 1. The map $\sin : \mathbb{R} \to \mathbb{R}$ is not surjective nor injective.



A bijective map from A to B

2. The map $\sin : [-\pi/2, \pi/2] \to \mathbb{R}$ is injective but not surjective.
3. The map $\sin : \mathbb{R} \to [-1, 1]$ is a surjective map. It is not injective.
4. The map $\sin : [-\pi/2, \pi/2] \to [-1, 1]$ is a bijective map.

Theorem 2.2.3 [Pigeonhole Principle] *Let $f : A \to B$ be a map between two sets of size $n \in \mathbb{N}$. Then f is injective if and only if it is surjective.*

Remark 2.2.4 The above result is called the pigeonhole principle because of the following. If one has n pigeons (the set A) and the same number of holes (the set B), then one pigeonhole is empty if and only if one of the other holes contains at least two pigeons.

Example 2.2.5 Suppose p and q are two distinct prime numbers. We consider the map $\phi : \{0, 1, \ldots, p-1\} \to \{0, 1, \ldots, p-1\}$ defined by $\phi(x) = y$ where y is the unique element in $\{0, 1, \ldots, p-1\}$ with $y = q \cdot x \mod p$.

We claim that the map ϕ is a bijection. By the pigeon hole principle it suffices to show that ϕ is injective.

So, let x, x' be two elements with $\phi(x) = \phi(x')$. Then $q \cdot x \mod p = q \cdot x' \mod p$ from which we deduce that $q \cdot (x - x') = 0 \mod p$. Since p is a prime distinct from q, we find $p \mid x - x'$. (We will go deeper into this in Chapter 2 of [1].) But then $x = x'$. Hence ϕ is injective and thus also bijective.

If $f : A \to B$ is a bijection, i.e., a bijective map, then for each $b \in B$ we can find a unique $a \in A$ with $f(a) = b$. So, also the relation $f^\top = \{(b, a) \in B \times A \mid (a, b) \in f\}$ is a map. This map is called the *inverse map* of f and denoted by f^{-1}.

Proposition 2.2.6 *Let $f : A \to B$ be a bijection. Then for all $a \in A$ and $b \in B$ we have $f^{-1}(f(a)) = a$ and $f(f^{-1}(b)) = b$. In particular, f is the inverse of f^{-1}.*

Proof. Let $a \in A$. Then $f^{-1}(f(a)) = a$ by definition of f^{-1}. If $b \in B$, then, by surjectivity of f, there is an $a \in A$ with $b = f(a)$. So, by the above, $f(f^{-1}(b)) = f(f^{-1}(f(a))) = f(a) = b$. □

Theorem 2.2.7 *Let $f : A \to B$ and $g : B \to C$ be two maps.*

1. *If f and g are surjective, then so is $g \circ f$;*
2. *If f and g are injective, then so is $g \circ f$;*
3. *If f and g are bijective, then so is $g \circ f$.*

Proof.

1. Let $c \in C$. By surjectivity of g there is a $b \in B$ with $g(b) = c$. Moreover, since f is surjective, there is also an $a \in A$ with $f(a) = b$. In particular, $g \circ f(a) = g(f(a)) = g(b) = c$. This proves $g \circ f$ to be surjective.
2. Let $a, a' \in A$ with $g \circ f(a) = g \circ f(a')$. Then $g(f(a)) = g(f(a'))$ and by injectivity of g we find $f(a) = f(a')$. Injectivity of f implies $a = a'$. This shows that $g \circ f$ is injective.
3. (i) and (ii) imply (iii).

□

Proposition 2.2.8 *If $f : A \to B$ and $g : B \to A$ are maps with $f \circ g = I_B$ and $g \circ f = I_A$, where I_A and I_B denote the identity maps on A and B, respectively. Then f and g are bijections. Moreover, $f^{-1} = g$ and $g^{-1} = f$.*

Proof. Let $b \in B$, then $f(g(b)) = b$. Thus the map f is surjective. If $a, a' \in A$ with $f(a) = f(a')$, then $a = g(f(a)) = g(f(a')) = a'$. Hence f is also injective. In particular, f is bijective. By symmetry we also find g to be bijective, and it follows that $f^{-1} = g$ and $g^{-1} = f$. □

Lemma 2.2.9 *Suppose $f : A \to B$ and $g : B \to C$ are bijective maps. Then the inverse of the map $g \circ f$ equals $f^{-1} \circ g^{-1}$.*

Proof. $(f^{-1} \circ g^{-1})(g \circ f)(a) = f^{-1}(g^{-1}(g(f(a)))) = f^{-1}(f(a)) = a$. □

2.3. Exercises

Exercise 2.3.1 Which of the following relations are maps from $A = \{1, 2, 3, 4\}$ to A?

1. $\{(1, 3), (2, 4), (3, 1), (4, 2)\}$;
2. $\{(1, 3)(2, 4)\}$;
3. $\{(1, 1), (2, 2), (3, 3), (4, 4), (1, 3), (2, 4), (3, 1), (4, 2)\}$;
4. $\{(1, 1), (2, 2), (3, 3), (4, 4)\}$.

Exercise 2.3.2 Suppose f and g are maps from $\mathbb{R}$ to $\mathbb{R}$ defined by $f(x) = x^2$ and $g(x) = x + 1$ for all $x \in R$. What is $g \circ f$ and what is $f \circ g$?

Exercise 2.3.3 Which of the following maps is injective, surjective or bijective?

1. $f : \mathbb{R} \to \mathbb{R}$, $f(x) = x^2$ for all $x \in \mathbb{R}$.
2. $f : \mathbb{R} \to \mathbb{R}_{\geq 0}$, $f(x) = x^2$ for all $x \in \mathbb{R}$.
3. $f : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$, $f(x) = x^2$ for all $x \in \mathbb{R}$.

Exercise 2.3.4 Suppose R_1 and R_2 are relations on a set S with $R_1; R_2 = I$ and $R_2; R_1 = I$. Prove that both R_1 and R_2 are bijective maps.

Exercise 2.3.5 Let R be a relation from a finite set S to a finite set T with adjacency matrix A. Prove the following statements:

1. If every row of A contains one nonzero entry, then R is a map.
2. If moreover, every column contains at most one entry, then the map R is injective.
3. If every row and column contain only one 1, then R is a bijection. What is the adjacency matrix of the inverse map?

Exercise 2.3.6 Let S and T be two sets. If R is a relation of $S \times T$, then for each $t \in T$ we have the pre-image

$$_R[t] = \{s \in S \mid sRt\}$$

which is a subset of S.

Prove that the relation $\{(t, _R[t]) \mid t \in T\}$ is a map from T to the power set $\mathcal{P}(S)$ of S.

Moreover, show that, if $f : T \to \mathcal{P}(S)$ is a map, then $R_f = \{(s, t) \mid s \in f(t)\}$ is a relation on $S \times T$ with $_{R_f}[t] = f$.

3. Order relations

3.1. Posets

Definition 3.1.1 A relation $\sqsubseteq$ on a set P is called an *order* if it is reflexive, antisymmetric and transitive. That means that for all x, y and z in P we have:

- $x \sqsubseteq x$;
- if $x \sqsubseteq y$ and $y \sqsubseteq x$, then $x = y$;
- if $x \sqsubseteq y$ and $y \sqsubseteq z$, then $x \sqsubseteq z$.

The pair $(P, \sqsubseteq)$ is called a *partially ordered set*, or for short, a *poset*.

Two elements x and y in a poset $(P, \sqsubseteq)$ are called comparable if $x \sqsubseteq y$ or $y \sqsubseteq x$. The elements are called incomparable if $x \not\sqsubseteq y$ and $y \not\sqsubseteq x$.

If any two elements $x, y \in P$ are comparable, so we have $x \sqsubseteq y$ or $y \sqsubseteq x$, then the relation is called a *linear* order.

Example 3.1.2

- The identity relation I on a set P is an order.
- On the set of real numbers $\mathbb{R}$ the relation $\leq$ is an order relation. For any two numbers $x, y \in \mathbb{R}$ we have $x \leq y$ or $y \leq x$. This makes $\leq$ into a linear order. Restriction of $\leq$ to any subset of $\mathbb{R}$ is again a linear order.
- Let P be the power set $\mathcal{P}(X)$ of a set X, i.e., the set of all subsets of X. Inclusion $\subseteq$ defines a partial order on P. This poset contains a smallest element $\emptyset$ and a largest element X. Clearly, $\subseteq$ defines a partial order on any subset of P.
- The relation "Is a divisor of" $\mid$ defines an order on the set of natural numbers $\mathbb{N}$. We can associate this example to the previous one in the following way. For each $a \in \mathbb{N}$ denote by $D(a)$ the set of all divisors of a. Then we have

$$a \mid b \Leftrightarrow D(a) \subseteq D(b).$$

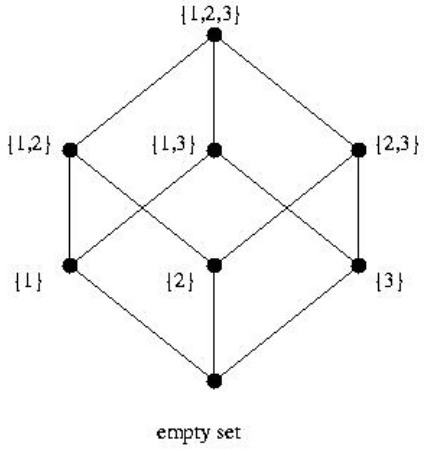
- On the set P of partitions of a set X we define the relation "refines" by the following. The partition Π_1 refines Π_2 if and only if each $\pi_1 \in \Pi_1$ is contained in some $\pi_2 \in \Pi_2$. The relation "refines" is a partial order on P.

  Notice, for the corresponding equivalence relations R_{Π_1} and R_{Π_2} we have Π_1 refines Π_2 if and only if $R_{\Pi_1} \subseteq R_{\Pi_2}$.
- If $\sqsubseteq$ is an order on a set P, then $\sqsupseteq$ also defines an order on P. Here $x \sqsupseteq y$ if and only if $y \sqsubseteq x$. The order $\sqsupseteq$ is called the *dual* order of $\sqsubseteq$.

Definition 3.1.3 If $\sqsubseteq$ is an order on the set P, then the corresponding directed graph with vertex set P and edges (x, y) where $x \sqsubseteq y$ is *acyclic* (i.e., contains no cycles of length > 1).

If we want to draw a picture of the poset, we usually do not draw the whole digraph. Instead we only draw an edge from x to y from P with $x \sqsubseteq y$ if there is no z, distinct from both x and y, for which we have $x \sqsubseteq z$ and $z \sqsubseteq y$. This digraph is called the *Hasse diagram* for $(P, \sqsubseteq)$, named after the german mathematician Helmut Hasse (1898-1979). Usually pictures of Hasse diagrams are drawn in such a way that

two vertices x and y with $x \sqsubseteq y$ are connected by an edge going upwards. For example the Hasse diagram for the poset $(\{1, 2, 3\}, \subseteq)$ is drawn as below.



3.1.4 [New posets from old ones] There are various ways of constructing new posets out of old ones. We will discuss some of them. In the sequel both P and Q are posets with respect to some order, which we usually denote by $\sqsubseteq$, or, if confusion can arise, by $\sqsubseteq_P$ and $\sqsubseteq_Q$.

- If P' is a subset of P, then P' is also a poset with order $\sqsubseteq$ restricted to P'. This order is called the *induced* order on P'.

- $\sqsupseteq$ induces the *dual* order on P.
- Let S be some set. On the set of maps from S to P we can define an ordering as follows. Let $f : S \to P$ and $g : S \to P$, then we define $f \sqsubseteq g$ if and only if $f(s) \sqsubseteq g(s)$ for all $s \in S$.
- On the cartesian product $P \times Q$ we can define an order as follows. For $(p_1, q_1), (p_2, q_2) \in P \times Q$ we define $(p_1, q_1) \sqsubseteq (p_2, q_2)$ if and only if $p_1 \sqsubseteq p_2$ and $q_1 \sqsubseteq q_2$. this order is called the *product order*.
- A second ordering on $P \times Q$ can be obtained by the following rule. For $(p_1, q_1), (p_2, q_2) \in P \times Q$ we define $(p_1, q_1) \sqsubseteq (p_2, q_2)$ if and only if $p_1 \sqsubseteq p_2$ and $p_1 \neq p_2$ or if $p_1 = p_2$ and $q_1 \sqsubseteq q_2$. This order is called the *lexicographic order* on $P \times Q$.

  Of course we can now extend this to direct products of more than two sets.

3.2. Maximal and Minimal Elements

Definition 3.2.1 Let $(P, \sqsubseteq)$ be a partially order set and $A \subseteq P$ a subset of P. An element $a \in A$ is called the *largest element* or *maximum* of A, if for all $a' \in A$ we have $a' \sqsubseteq a$. Notice that a maximum is unique, see Lemma 3.2.2 below.

An element $a \in A$ is called *maximal* if for all $a' \in A$ we have that either $a' \sqsubseteq a$ or a and a' are incomparable.

Similarly we can define the notion of *smallest element* or *minimum* and *minimal element*.

If the poset $(P, \sqsubseteq)$ has a maximum, then this is often denoted as $\top$ (top). A smallest element is denoted by $\bot$ (bottom).

If a poset $(P, \sqsubseteq)$ has a minimum $\bot$, then the minimal elements of $P \setminus \{\bot\}$ are called the *atoms* of P.

Lemma 3.2.2 *Let $(P, \sqsubseteq)$ be a partially order set. Then P contains at most one maximum and one minimum.*

Proof. Suppose $p, q \in P$ are maxima. Then $p \sqsubseteq q$ as q is a maximum. Similarly $q \sqsubseteq p$ as p is a maximum. But then by antisymmetry of $\sqsubseteq$ we have $p = q$. □

Example 3.2.3

- If we consider the poset of all subsets of a set S, then the empty set $\emptyset$ is the minimum of the poset, whereas the whole set S is the maximum. The atoms are the subsets of S containing just a single element.
- If we cosider $|$ as an order on $\mathbb{N}$, then 1 is the minimal element and 0 the maximal element. The atoms are those natural numbers > 1, that are only divisible by 1 and itself, i.e., the prime numbers.

Lemma 3.2.4 *Let $(P, \sqsubseteq)$ be a finite poset. Then P contains a minimal and a maximal element.*

Proof. Consider the directed graph associated to $(P, \sqsubseteq)$ and pick a vertex in this graph. If this vertex is not maximal, then there is an edge leaving it. Move along this edge to the neighbor. Repeat this as long as no maximal element is found. Since the graph contains no cycles, we will never meet a vertex twice. Hence, as P is finite, the procedure has to stop. This implies we have found a maximal element.

A minimal element of $(P, \sqsubseteq)$ is a maximal element of $(P, \sqsupseteq)$ and thus exists also. □

Example 3.2.5 Notice that minimal elements and maximal elements are not necessarily unique. In fact, they do not even have to exist. In $(\mathbb{R}, \leq)$ for example, there is no maximal nor a minimal element.

Algorithm 3.2.6 [Topological sorting] Given a finite poset $(P, \sqsubseteq)$, we want to sort the elements of P in such a way that an element x comes before an element y if $x \sqsubseteq y$. This is called *topological sorting*. In other words, topological sorting is finding a map $\mathsf{ord} : P \to \{1, \ldots, n\}$, where $n = |P|$, such that for distinct x and y we have that $x \sqsubseteq y$ implies $\mathsf{ord}(x) < \mathsf{ord}(y)$. We present an algorithm for topological sorting.

Suppose we are given a finite poset $(P, \sqsubseteq)$, then for each element p in P we determine the indegree, i.e., the number of elements q with $q \sqsubseteq p$. While there are vertices in P with indegree 0, pick one of them, say q, and set $\mathsf{ord}(q)$ to be the smallest value in $\{1, \ldots, n\}$ which is not yet an image of some point. Now remove q from P and lower all the indegrees of the neighbors of q by 1.

Notice that, by Lemma 3.2.4, we will always find elements in P with indegree 0, unless P is empty.

Example 3.2.7 Topological sort has various applications. For example consider a spreadsheet. In a spreadsheet various tasks depend on each other. In particular, some of the computations need input from other computations and therefore they can only be carried out after completion of the other computations. If there are no cycles in these computations, this puts a partial order on the set of tasks within a spreadsheet. By topological sort the task list can be linearized and the computations can be done in a linear order.

Definition 3.2.8 If $(P, \sqsubseteq)$ is a poset and $A \subseteq P$, then an *upperbound* for A is an element u with $a \sqsubseteq u$ for all $a \in A$.

A *lowerbound* for A is an element u with $u \sqsubseteq a$ for all $a \in A$.

If the set of all upperbounds of A has a minimal element, then this element is called the *largest upperbound* or *supremum* of A. Such an element, if it exists, is denoted by $\sup A$. If the set of all lowerbounds of A has a maximal element, then this element

is called the *least lowerbound* or *infimum* of A. If it exists, the infimum of A is denoted by $\inf A$.

Example 3.2.9 Let S be a set. In $(\mathcal{P}(S), \subseteq)$ any set A of subsets of S has a supremum and an infimum. Indeed,

$$\sup A = \bigcup_{X \in A} X \text{ and } \inf A = \bigcap_{X \in A} X.$$

Example 3.2.10 If we consider the poset $(\mathbb{R}, \leq)$, then not every subset A of $\mathbb{R}$ has a supremum or infimum. Indeed, $\mathbb{Z} \subseteq \mathbb{R}$ has no supremum an no infimum.

Example 3.2.11 In $(\mathbb{N}, |)$ the supremum of two elements a and b is the least common multiple of a and b. Its infinum is the greatest common divisor.

If $(P, \sqsubseteq)$ is a finite poset, then as we have seen above, we can order the elements from P as $p_1, p_2, \ldots, p_n$ such that $p_i \sqsubseteq p_j$ implies $i < j$. This implies that the adjacency matrix of $\sqsubseteq$ is uppertriangular, which means that it has only nonzero entries on or above the main diagonal.

Definition 3.2.12 An *ascending chain* in a poset $(P, \sqsubseteq)$ is a (finite or infinite) sequence $p_0 \sqsubseteq p_1 \sqsubseteq \ldots$ of elements p_i in P. A *descending chain* in $(P, \sqsubseteq)$ is a (finite or infinite) sequence of elements p_i, $i \geq 0$ with $p_0 \sqsupseteq p_1 \sqsupseteq \ldots$ of elements p_i in P.

The poset $(P, \sqsubseteq)$ is called *well founded* if any descending chain is finite.

Example 3.2.13 The natural numbers $\mathbb{N}$ with the ordinary ordering $\leq$ is well founded. Also the ordering $|$ on $\mathbb{N}$ is well founded.

However, on $\mathbb{Z}$ the order $\leq$ is not well founded.

3.3. Exercises

Exercise 3.3.1 Let $|$ denote the relation "is a divisor of " defined on $\mathbb{Z}$. Even if we let 0 be a divisor of 0, then this does not define an order on $\mathbb{Z}$. Prove this.

Exercise 3.3.2 Let $|$ denote the relation "is a divisor of " . This relation defines an order on the set $D = \{1, 2, 3, 5, 6, 10, 15, 30\}$ of divisors of 30. Draw the Hasse diagram.

Draw also the Hasse diagram of the poset of all subsets of $\{2, 3, 5\}$. Compare the two diagrams. What do you notice?

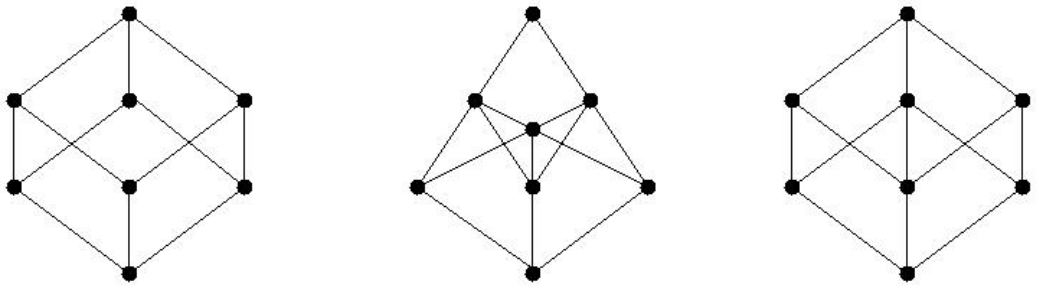
Exercise 3.3.3 Let $\sqsubseteq$ denote an order relation on a finite set P. By H we denote the relation defining adjacency in the Hasse diagram of $\sqsubseteq$. Prove that $\sqsubseteq$ is the transitive reflexive closure of H.

Exercise 3.3.4 Let $m, n \in \mathbb{N}$. By Π_m we denote the partition of $\mathbb{Z}$ into equivalence classes modulo m. What is a necessary and sufficient condition on n and m for Π_m to be a refinement of Π_n.

Exercise 3.3.5 Suppose $\sqsubseteq_P$ and $\sqsubseteq_Q$ are linear orders on P and Q, respectively. Show that the lexicographical order on $P \times Q$ is also linear.

Exercise 3.3.6 Show that the relations as defined in 3.1.4 are indeed orders.

Exercise 3.3.7 In the figure below you see three diagrams. Which of these diagrams are Hasse diagrams?



Exercise 3.3.8 Consider $\mathbb{N}_{>0}$ the set of positive integers. a partition of n is a multi-set of positive integers such that the sum of all elements in this multi-set equals n.

For example $\{1, 1, 3, 5\}$ is a partition of 10, and so are $\{10\}$ and $\{5, 5\}$. We call a multi-set M less than or equal to a Multiset M' if M is obtained by partitioning 0 or some members of M'. So, $\{1, 1, 2, 2, 3\}$, $\{1, 1, 1, 1, 1, 1, 4\}$ and $\{2, 2, 2, 3\}$ are less than $\{4, 5\}$.

a) Show that "less than or equal to" for multi-sets is an order.

b) Determine all partitions of 6 and draw the Hasse diagram on all these partitions.

Exercise 3.3.9 Suppose $(A, \sqsubseteq_A)$ and $(B, \sqsubseteq_B)$ are posets. If A and B are disjunct, then we define the relation $\sqsubseteq$ o $A \cup B$ as follows:

$$\begin{aligned} x \sqsubseteq y \text{ if} \quad & x, y \in A \text{ and } x \sqsubseteq_A y; \\ \text{or} \quad & x, y \in B \text{ and } x \sqsubseteq_B y; \\ \text{and if} \quad & x \in A \text{ and } y \in B. \end{aligned}$$

a) Prove that $\sqsubseteq$ is an order on $A \cup B$.

b) Give necessary and sufficient conditions such that $\sqsubseteq$ is a linear order on $A \cup B$.

Exercise 3.3.10 On $\mathbb{Z}$ we define $\sqsubseteq$ by $x \sqsubseteq y$ if and only if $x - y$ is odd and x is even, or, $x - y$ is even and $x \leq y$.

Show that $\sqsubseteq$ is an order on $\mathbb{Z}$. Is this order linear?

4. Recursion and Induction

4.1. Recursion

A *recursive definition* tells us how to build objects by using ones we have already built. Let us start with some examples of some common functions from $\mathbb{N}$ to $\mathbb{N}$ which can be defined recursively:

Example 4.1.1 The function $f(n) = n!$ can be defined recursively:

$$\begin{array}{rrcl} & f(0) & := & 1; \\ \text{for } n > 0\text{:} & f(n) & := & n \cdot f(n-1). \end{array}$$

Example 4.1.2 The sum $1+2+\cdots+n$ can also be written as $\sum_{i=1}^{n} i$. Here we make use of the summation symbol $\sum$, which, for any map f with domain $\mathbb{N}$, we recursively define by:

$$\begin{array}{rrcl} & \sum_{i=1}^{1} f(i) & := & f(1); \\ \text{for } n > 1\text{:} & \sum_{i=1}^{n} f(i) & := & [\sum_{i=1}^{n-1} f(i)] + f(n); \end{array}$$

Similarly, $n!$ is often expressed as $\prod_{i=1}^{n} i$. Here we use the product symbol $\prod$ which is recursively defined by:

$$\begin{array}{rrcl} & \prod_{i=1}^{1} f(i) & := & f(1); \\ \text{for } n > 1\text{:} & \prod_{i=1}^{n} f(i) & := & [\prod_{i=1}^{n-1} f(i)] \cdot f(n); \end{array}$$

Example 4.1.3 [Fibonacci sequence] The Italian mathematician Fibonacci (1170-1250) studied the population growth of rabbits. He considered the following model of growth. Start with one pair of rabbits, one male and one female rabbit.

As soon as a pair of rabbits, male and female, is one months old, it starts producing new rabbits. It takes another month before the young rabbits, again a pair consisting of a male and a female rabbit, are born. Let $F(n)$ denote the number of pairs in month n. We have the following recursive definition for F. Here $n \in \mathbb{N}$:

$$\begin{aligned} F(1) &:= 1; \\ F(2) &:= 1; \\ F(n+2) &:= F(n+1) + F(n). \end{aligned}$$

Indeed, in month $n+2$ we still have the pairs of one month earlier, i.e., $F(n+1)$, but also the young pairs of those pairs which are at least one month old in month $n+1$, i.e., the number of pairs in month n.

In the examples above we see that for a recursively defined function f we need two ingredients:

- a *base* part, where we define the function value $f(n)$ for some small values of n like 0 or 1.
- a recursive part in which we explain how to compute the function in n with the help of the values for integers smaller than n.

Of course, we do not have to restrict our attention to functions with domain $\mathbb{N}$. Recursion can be used at several places.

Example 4.1.4 Let S be the subset of $\mathbb{Z}$ defined by:

$3 \in S$;
if $x, y \in S$ then also $-x$ and $x + y \in S$.

Then S consists of all the multiples of 3. Indeed, if $n = 3m$ for some $m \in \mathbb{N}$, then $n = (\ldots(3+3)+3)+\cdots+3)+3$, and hence is in S. But then also $-3m \in S$. Thus S contains all multiples of 3.

On the other hand, if S contains only multiples of 3, then in the next step of the recursion, only multiples of 3 are added to S. So, since initially S contains only 3, S contains only multiples of 3.

Example 4.1.5 Suppose R is a relation on a set S. We define $\overline{R} \subseteq S \times S$ recursively by

$R \subseteq \overline{R}$
if (a, b) and (b, c) in $\overline{R}$ then also $(a, c) \in \overline{R}$.

Then $\overline{R}$ is the transitive closure of R. Indeed, $\overline{R}$ contains R and is transitive. Hence it contains the transitive closure of R. We only have to show that $\overline{R}$ is contained in the transitive closure of R. This will be shown 4.5.2.

Example 4.1.6 Suppose Σ is a set of symbols. By Σ^* we denote the set of all strings over Σ. The set Σ^* can be defined by the following:

λ (the empty string) is in Σ^*;
if $w \in \Sigma^*$ and $s \in \Sigma$, then $w.s$ is in Σ^*.

Here . stands for concatenation of the strings. So, If $\Sigma = \{a, , b, c\}$, then

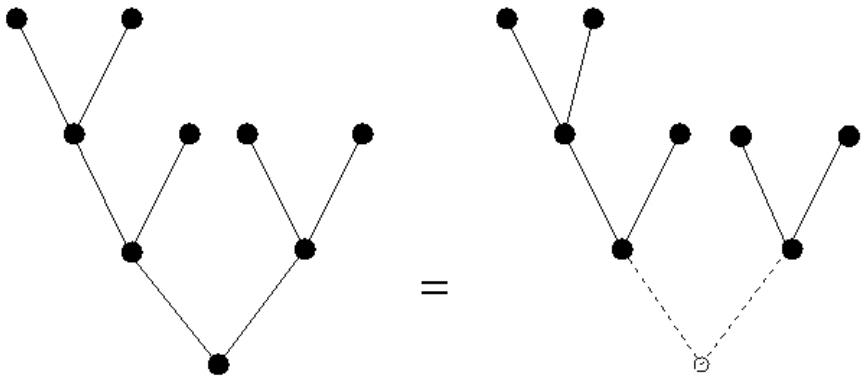
$$\Sigma^* = \{\lambda, a, b, c, aa, ab, ac, ba, bb, bc, ca, cb, cc, aaa, \dots\}.$$

Example 4.1.7 A (finite, directed) *tree* is a (finite) digraph Γ such that:

- Γ contains no cycles;
- there is a unique vertex, called the *root* of the tree with indegree 0; all other vertices have indegree 1;
- for any vertex v there is a path from the root to v.

A tree is called *binary* is every vertex has outdegree 0 or 2. Notice that the graph consisting of a single vertex is a binary tree.

Moreover, if $T_1 = (V_1, E_1)$ and $T_2 = (V_2, E_2)$ are binary trees, then we can make a new binary tree $\mathsf{Tree}(T_1, T_2)$ in the following way. As vertex set we take the vertices of T_1 and T_2 and add a new vertex r. This vertex r is the root of the new tree and is the tail of two new edges with head r_1 and r_2, the roots of T_1 and T_2, respectively. All other edges come from $T-1$ and T_2. So $\mathsf{Tree}(T_1, T_2) = (V_1 \cup V_2 \cup \{r\}, E_1 \cup E_2 \cup \{(r, r_1), (r, r_2)$.



We can also give a recursive definition of the set of finite binary trees in the following way.

The set $\mathcal{T}$ of finitary trees is defined by:

- the binary tree on a single vertex in in $\mathcal{T}$;
- if T_1 and T_2 are in $\mathcal{T}$, then $\mathsf{Tree}(T_1, T_2)$ is in $\mathcal{T}$.

Notice that a recursive definition of some operation or structure consists of:

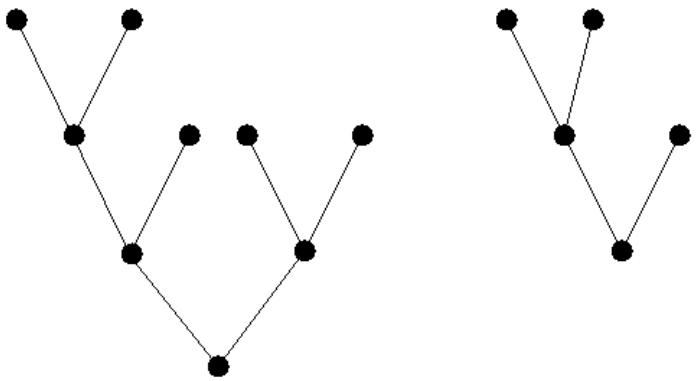
- a definition of the basic structures or operations
- a procedure to construct new basic structures or operations out of already constructed ones.

These two ingredients do not guarantee that a recursion is well defined. To avoid contradicting rules, we assume that if an object x is used (at some stage) in the construction of an object y, then y is not used in the construction of x.

This leads to an ordering $\sqsubseteq$ on the objects constructed. The basic objects are the minimal elements of the order; if $x_1, \ldots, x_n$ are objects used to create y then we say $x_i \sqsubset y$. The transitive and reflexive closure $\sqsubseteq$ of this relation is an order.

Indeed, if $x \sqsubseteq y$, then x is used in the construction of y but, unless $x = y$, the object y is not used for constructing x. As each object is constructed in finitely many steps, the order $\sqsubseteq$ only has descending chains of finite length. It is well founded.

Example 4.1.8 Consider the set $\mathcal{T}$ of finite directed binary trees as defined in Example 4.1.7. If $T_i = (V_i, E_i)$, $i = 1, 2$, are trees in $\mathcal{T}$, then we say $T_1 \sqsubseteq T_2$ if and only if $V_1 \subseteq V_2$ and E_1 is the set of all edges in E_2 with tail in V_1.



This relation is a well founded order on $\mathcal{T}$. (Prove this!) It is the transitive closure of the relation $\sqsubset$ defined above.

4.2. Linear recurrences

We now consider a special class of recurrence relations. A *linear* recursion is a function $F : \mathbb{N} \to \mathbb{C}$ satisfying the following recursion:

$$F(n) = a_1F(n-1) + a_2F(n-2) + \cdots + a_kF(n-k)$$

for some fixed $a_1, \ldots, a_k \in \mathbb{C}$.

In various cases it is possible to give a closed formula for the function F. First an example.

Example 4.2.1 (Fibonnaci) Consider the recursive fuction $F : \mathbb{N} \to \mathbb{N}$ given by

$$F(n+2) = F(n+1) + F(n).$$

The Fibonacci sequence is an example of a function satisfying this linear recurrence, but there are more. If $F(x) = \alpha^n$ for some α, then we deduce from the recursion that

$$\alpha^2 = \alpha + 1.$$

So, for $\alpha = \frac{1+\sqrt{5}}{2}$ and $\alpha = \frac{1-\sqrt{5}}{2}$ we have found two more examples satisfying the recursion. Moreover, combinining the latter two examples we get a whole family of fucntions satisfying the recursion. Indeed, for all $\lambda, \mu \in \mathbb{C}$ we see that

$$F(n) := \lambda(\frac{1+\sqrt{5}}{2})^n + \mu(\frac{1-\sqrt{5}}{2})^n$$

satisfies the recursion.

The Fibonocci sequence is among these functions, we have to set both λ and μ equal to 1. Actually by varying λ and μ we obtain all functions F satisfying the recursion. This is clear from the following. A function F satisfying the above recurrence relation is completely determined by the values of F at 1 and 2. Taking for λ and μ a solution to the system

$$F(1) = \lambda(\frac{1+\sqrt{5}}{2}) + \mu(\frac{1-\sqrt{5}}{2})$$

$$F(2) := \lambda(\frac{1+\sqrt{5}}{2})^2 + \mu(\frac{1-\sqrt{5}}{2})^2$$

have found a way to express F as

$$F(n) := \lambda(\frac{1+\sqrt{5}}{2})^n + \mu(\frac{1-\sqrt{5}}{2})^n$$

The method as described in the above example works for all linear recurrences.

If F is a function satisfying the recursion

$$F(n) = a_1F(n-1) + a_2F(n-2) + \cdots + a_kF(n-k)$$

for some fixed $a_1, \ldots, a_k \in \mathbb{R}$, then we can try $F(n) = \alpha^n$ for F.

This will work if and only if

$$\alpha^k = a_1\alpha^{k-1} + a_2\alpha^{n-2} \cdots + a_{k-1}\alpha + a_k.$$

So, α is a root of the polynomial equation

$$x^k = a_1x^{k-1} + \cdots + a_{k-1}x + a_k.$$

This equation is called the characteristic polynomial equation related to the recursion. In general, there are k (complex) solutions to this equation, $\alpha_1, \ldots, \alpha_k$ yielding k functions $F_i(n) = \alpha_i^n$ satisfying the recursion. If all roots $\alpha_1, \ldots, \alpha_k$ are distinct, then the functions F_i are linearly independent. This implies that there are unique $\lambda_1, \ldots, \lambda_k$ with $F(i) = \lambda_1\alpha_1^i + \cdots + \lambda_k\alpha_k^i$ for $i = 1, \ldots, k$. So, if F satifies the recursion, then F is uniquely determined by the values $F(1), \ldots, F(k)$ and

$$F(n) = \lambda_1F_1(n) + \cdots + \lambda_kF_k(n) = \lambda_1\alpha_1^n + \cdots + \lambda_k\alpha_k^n,$$

for all $n \in \mathbb{N}$.

If not all α_i are distinct, we have some additional solutions to the recursion. Suppose α is a root of multiplicity d of the equation

$$x^k = a_1x^{k-1} + a_2x^{n-2} \cdots + a_{k-1}x + a_k.$$

Then not only α^n satisfies the recursion but also $n\alpha^n, \ldots, n^{d-1}\alpha^n$. Again we find enough independent solutions to form all solutions to the recursion.

Example 4.2.2 Consider the recursion $F(n) = 3F(n-2) - 2F(n-3)$. The characteristic polynomial related to the recursion equals $x^3 = 3x - 2$, whch can be written as

$$(x-1)^2(x-2) = 0.$$

So, the roots of the equation are 1 (with multiplicity 2) and 2. Three function satisfying the recursion are then $F_1(n) = 1^n$, $F_2(n) = n \cdot 1^n = n$ and $F_3(n) = n^2$. So, the arbitrary solution to the recurcence is given by

$$F(n) = \lambda + \mu n + \rho n^2,$$

where λ, μ and ρ are some real or complex numbers.

4.3. Natural Induction

4.3.1 Principle of Natural Induction. Suppose $P(n)$ is a predicate for $n \in \mathbb{Z}$. Let $b \in \mathbb{Z}$. If the following holds:

- $P(b)$ is true;
- for all $k \in \mathbb{Z}, k \geq b$ we have that $P(k)$ implies $P(k+1)$.

Then $P(n)$ is true for all $n \geq b$.

We give some examples:

Example 4.3.2 We claim that for all $n \in \mathbb{N}$ the sum

$$\sum_{i=1}^{n} i = \frac{1}{2}n(n+1).$$

We first check the claim for $n = 1$:

$$\sum_{i=1}^{1} i = 1 = \frac{1}{2}1(1+1).$$

Now suppose that for some $k \in \mathbb{N}$ we do have

$$\sum_{i=1}^{k} i = \frac{1}{2}k(k+1).$$

Then

$$\sum_{i=1}^{k+1} i = (\sum_{i=1}^{k} i) + (k+1) = \frac{1}{2}k(k+1) + (k+1) = \frac{1}{2}(k+1)(k+2).$$

Hence if the claim holds for some k in $\mathbb{N}$, then it also holds for $k+1$.
The principle of natural Induction implies now that for all $n \in \mathbb{N}$ we have

$$\sum_{i=1}^{n} i = \frac{1}{2}n(n+1).$$

Example 4.3.3 For all $n \in \mathbb{N}$ and $x \in \mathbb{R}$, $x \neq 1$, we have

$$\sum_{i=1}^{n} x^i = \frac{x^{n+1} - x}{x - 1}.$$

Here is a proof of this statement using natural induction. First consider the case $n = 1$. Then the left hand side of the above equation equals x. The right hand side equals $\frac{x^2-x}{x-1} = x$. So, for $n = 1$, equality holds.

Now assume that $\sum_{i=1}^{k} x^i = \frac{x^{k+1}-x}{x-1}$ for some $k \in \mathbb{N}$. Then $\sum_{i=1}^{k+1} x^i = [\sum_{i=1}^{k} x^i] + x^{k+1}$. By assumption this equals $\frac{x^{k+1}-x}{x-1} + x^{k+1} = \frac{x^{k+2}-x}{x-1}$.

The Principle of Natural Induction implies now that for all $n \in \mathbb{N}$ we have

$$\sum_{i=1}^{n} x^i = \frac{x^{n+1} - x}{x - 1}.$$

Example 4.3.4 Let $a, b, c \in \mathbb{R}$. A *linear recurrence* is a recurrence relation of the form

$$\begin{array}{rcl} a_0 & := & a; \\ a_{n+1} & := & b \cdot a_n + c; \end{array}$$

This is a generalization of the the recurrence relation as given in Example 4.3.3. For linear recurrence relations we can find a closed formula. Indeed,

$$a_n = b^n \cdot a + b^{n-1}c + b^{n-1} \cdot c + \cdots + b \cdot c + c = b^n \cdot a + \left(\frac{b^n - 1}{b - 1}\right) \cdot c.$$

We give a proof by induction.

For $n = 1$ we indeed have $a_1 = b \cdot a + c = b^1 \cdot a + \left(\frac{b^1-1}{b-1}\right) \cdot c$.

Suppose that for some $k \in \mathbb{N}$ we do have the equality

$$a_k = b^k \cdot a + \left(\frac{b^k - 1}{b - 1}\right) \cdot c.$$

Then

$$\begin{array}{rl} a_{k+1} & = b \cdot a_k + c \\ & = b \cdot (b^k \cdot a + \left(\frac{b^k-1}{b-1}\right) \cdot c) + c \\ & = b^{k+1} \cdot a + \left(\frac{b^{k+1}-b}{b-1}\right) \cdot c + c \\ & = b^{k+1} \cdot a + \left(\frac{b^k-b+(b-1)}{b-1}\right) \cdot c \\ & = b^{k+1} \cdot a + \left(\frac{b^k-1)}{b-1}\right) \cdot c. \end{array}$$

By the principle of natural induction we now have proven that

$$a_n = b^n \cdot a + \left(\frac{b^n - 1}{b - 1}\right) \cdot c.$$

for all $n \in \mathbb{N}$ with $n > 0$.

Example 4.3.5 Let S be a set with n elements, then $\mathcal{P}(S)$, the set of all subsets of S has size 2^n. We give a proof by induction.

For $n = 0$, the set S is the empty set and S itself is the only subset of S. So indeed, in this case $\mathcal{P}(S)$ has size $2^0 = 1$.

Suppose for some $k \in \mathbb{N}$ all sets of size k have exactly 2^k distinct subsets. Then consider a set S of size $k+1$. Fix an element $s \in S$. Then all subsets of S not containing s are precisely the subsets of $S \setminus \{s\}$. Hence, there are 2^k such subsets of S. For each such subset T there is a unique subset $T \cup \{s\}$ of S containing s. As every subset T' of S containing s is obtained as $T' \setminus \{s\} \cup \{s\}$ there are also 2^k subsets containing s.

We conclude that $\mathcal{P}(S)$ contains $2^k + 2^k = 2^{k+1}$ elements.

Now the principle of natural induction implies that every set S of n elements admits exactly 2^n subsets.

Example 4.3.6 Consider the function f defined by:
$f(0) = 1$;
$f(n+1) = \frac{1}{n^2+1} f(n)$, where $n \in \mathbb{N}$.

As we have seen in the above examples, a proof by natural induction consists of 4 steps:

- A statement $P(n)$ for all $n \in \mathbb{N}$.
- A base b, for which $P(b)$ is true.
- A proof that for all $k \in \mathbb{N}$ (or $k \geq b$) we have: $P(k)$ implies $P(k+1)$.
- The conclusion that for all $n \geq b$ we have $P(n)$ is true.

4.4. Strong Induction and Minimal Counter Examples

In this section we discuss two variations on Natural Induction. The first is strong induction.

4.4.1 Principle of Strong Induction. Suppose $P(n)$ is a predicate for $n \in \mathbb{Z}$. Let $b \in \mathbb{Z}$. If the following holds:

- $P(b)$ is true;
- for all $k \in \mathbb{Z}, k \geq b$ we have that $P(b), P(b+1), \ldots, P(k-1)$ and $P(k)$ together imply $P(k+1)$.

Then $P(n)$ is true for all $n \geq b$.

(Of course strong induction is just a variation of natural induction. Indeed, just replace the predicate $P(n)$ by the predicate $Q(n) := P(b) \wedge P(b+1) \wedge \cdots \wedge P(n)$.)

We give some examples.

Example 4.4.2 Consider the game of Nimm. In this game for two players a (positive) number of sticks is placed on the table. The two players take turns removing one, two or three sticks from the table. The player to remove the last stick form the table loses.

> The first player has a winning strategy if and only if the number of sticks, n say, is not of the form $4m + 1$, with $m \in \mathbb{N}$. Otherwise, the second player has a winning strategy.

We prove this statement with strong induction.

If $n = 1$, then the first player has to take the stick from the table and loses.

Now suppose that the statement is correct for all values of n with $1 \leq n \leq k$ for some $k \in \mathbb{N}$. We will prove it to be true for $n = k + 1$.

We divide the prove in two parts:

- $k + 1 = 4m + 1 > 1$ for some $m \in \mathbb{N}$.
  Since the first player can remove 1, 2 or 3 sticks, the second player is faced with k, $k - 1$ or $k - 2$ sticks. Since these numbers are not of the form $4l + 1$, $l \in \mathbb{N}$, our induction hypothesis implies that there is a winning strategy for the second player.

- $k + 1 = 4m + i$ for some $m \in \mathbb{N}$ and $i = 2, 3$ or 4.
  The first player can removes $i - 1$ sticks. Then the second player is facing $4m + 1$ sticks. By our induction hypothesis, there is a winning strategy for the first player.

Example 4.4.3 Suppose you have to divide an $n \times m$ chocolate bar into nm pieces. Then you will need to break it at least $nm - 1$ times.

This we can prove by strong induction. Suppose nm.

If $nm = 1$, then we are dealing with a single piece of chocolate, and we don't have to do anything. So indeed, we need zero breaks.

Suppose, $nm > 1$ and for all $n' \times m'$ bars with $n'm' < nm$, we need at least $n'm' - 1$ breaks to divided into $n'm'$ pieces. Then consider an $n \times m$ bar. Break it ones. Then one obtains two bars B_0 and B_1 of size $n_0 \times m_0$ and $n_1 \times m_1$, respectively, with $n_0m_0 + n_1m_1 = nm$. By our induction hypothesis, one has to break bar B_0 at least $n_0m_0 - 1$ times and bar B_0 at least $n_0m_0 - 1$ times. Hence in total we have to break the bar at least $1 + (n_0m_0 - 1) + (n_1m_1 - 1) = nm - 1$.

By the principle of strong induction we have shown that indeed one has to break an $n \times m$ chocolate bar at least $nm - 1$ times to get nm pieces.

The second variation of natural induction that we discuss is the (non)-existence of a minimal counter example.

4.4.4 Minimal Counter Example. Let $P(n)$ be a predicate for all $n \in \mathbb{Z}$. Let $b \in \mathbb{Z}$. If $P(n)$ is **not** true for all $n \in \mathbb{Z}, n \geq b$, then there is a minimal counter example. That means, there is an $m \in \mathbb{Z}, m \geq b$ with

$P(m)$ false and
$P(n)$ true for all $n \in \mathbb{N}$ with $b \leq n < m$.

Example 4.4.5 A prime is a a natural number $p > 2$ such that each divisor of p equals 1 or$p$. Every element $n \in \mathbb{N}$ with $n > 1$ is divisible by a prime.

Suppose m is a minimal counter example to this statement. Then, as $m|m$, we find that m cannot be prime. Hence, it admits a divisor $1 < m_1 < m$. As m is a minimal counter example to the statement, m_1 is divisible by some prime p. But by transitivity of the relation "divides", p also divides m. This contradicts m being the minimal counter example. Hence we have proved the statement.

4.5. Structural Induction

In this final section we discuss another variation of induction, the so-called *structural induction*. If a structure of data types is defined recursively, then we can use this recursive definition to derive properties by induction.

In particular,

- if all basic elements of a recursively defined structure satisfy some property P,
- and if newly constructed elements satisfy P, assuming the elements used in the construction already satisfy P,

then all elements in the structure satisfy P.

We give some examples.

Example 4.5.1 In Example 4.1.7 we have given a recursive definition of the set $\mathcal{T}$ of finite binary trees.

($*$) In every binary tree T the number edges is one less than the number of vertices.

We prove this by induction:

The tree consisting of a single vertex has 1 vertex and 0 edges. Hence for this tree the statement is correct.

Now assume suppose a tree $T = (V, E)$ is obtained as $\mathsf{Tree}(T_1, T_2)$ where $T_1 = (V_1, E_1)$ and $T_2 = (V_2, E_2)$ are two binary trees satisfying ($*$). Then the number of vertices in T equals $1 + |V_1| + |V_2|$, and the number of edges equals $2 + |E_1| + |E_2|$. Since $|V_1| = |E_i|+1$ we find that $|V| = |V_1|+|V_2|+1 = (|E_1|+1)+(|E_2|+1)+1 = |E|+1$. Hence T also satisfies ($*$).

This proves that all finite binary trees satisfy ($*$).

Example 4.5.2 Let R be a relation on a set S. In 4.1.5 we defined the relation $\overline{R}$. We will use structure induction to show that $\overline{R}$ is the transitive closure of R.

We already showed that $\overline{R}$ contains the transitive closure. So it remains to prove that $\overline{R}$ is contained in the closure.

Denote the transitive closure of R by TR. Our first step in the proof is to show that R is contained in TR. But this is by definition of TR. Next suppose $(a, b), (b, c)$ of $\overline{R}$ are also in TR, then the element (a, c) of $\overline{R}$ is also in TR as TR is transitive. Hence by structural induction, we have $\overline{R} \subseteq TR$ and hence we may conclude that $\overline{R} = TR$.

Although we will not go into the details, we want to mention that natural, strong and structural induction are actually a particular cases of induction on a well founded order:

4.5.3 The Principle of Induction on a well founded order. Let $(P, \sqsubseteq)$ be a well founded order. Suppose $Q(x)$ is a predicate for all $x \in P$ satisfying:

- $Q(b)$ is true for all minimal elements $b \in P$.
- If $x \in P$ and $Q(y)$ is true for all $y \in P$ with $y \sqsubseteq x$ but $y \neq x$, then $P(x)$ holds.

Then $Q(x)$ holds for all $x \in P$.

4.6. Exercises

Exercise 4.6.1 John wants to buy a new house. Therefore he needs \$200,000 from the bank. He can pay off this mortgage in 20 years, \$10,000 a year. Besides these \$10,000, John also has to pay 8% intrest a year over the amount, he is still has to pay to the bank.

What is the total amount John has to pay to the bank for this mortgage of \$200.000?

Exercise 4.6.2 Suppose $f(n)$ is the number of strings of length n with symbols from the alphabet $\{a, b, c, d\}$ with an even number of a's.

a) What is $f(0)$? And what $f(1)$?

b) Show that f satisfies the recurrence

$$f(n + 1) = 2 \cdot f(n) + 4^n.$$

Exercise 4.6.3 Suppose f satisfies the recurrence relation

$$f(n + 2) := 2f(n + 1) - 4f(n).$$

Show that for all $n \in \mathbb{N}$ we have $f(n + 3) = -8f(n)$.

Exercise 4.6.4 Let F be the Fibonacci sequence.

a) Show that for all $n > 2$ we have

$$F(n+2) = 1 + \sum_{i=1}^{n} F(i).$$

b) Show that for all $n > 2$ we have

$$F(2n+1) = 1 + \sum_{i=1}^{n} F(2i).$$

Exercise 4.6.5 Suppose f is defined on $\mathbb{N}$ by

$f(0) := 1,$
$f(n) = \frac{2n}{n+1} f(n-1)$ for all $n > 0$

Compute $f(1), f(2), \ldots, f(5)$. Can you find a closed formula for $f(n)$? Prove that your formula is correct for all $n \in \mathbb{N}$.

Exercise 4.6.6 Suppose f is defined on $\mathbb{N}$ by

$$\sum_{i=1}^{n} \frac{2i-1}{i^4 - 2i^3 + 3i^2 - 2i + 2}$$

Compute $f(1), f(2), \ldots, f(5)$. Can you find a closed formula for $f(n)$? Prove that your formula is correct for all $n \in \mathbb{N}$.

Exercise 4.6.7 Suppose f is defined on $\mathbb{N}$ by

$$\sum_{i=1}^{n} \frac{3i^2 - 3i + 1}{(i^3+1)(i^3 - 3i^2 + 3i)}$$

Compute $f(1), f(2), \ldots, f(5)$. Can you find a closed formula for $f(n)$? Prove that your formula is correct for all $n \in \mathbb{N}$.

Exercise 4.6.8 In a triangle in the plane, the sum of all the three angles equals $180°$. In a 4-gon, the sum of all the four angles equals $360°$. How about the sum of the angles in a convex n-gon with $n \geq 5$? (An n-gon is called convex, if any straight line between two vertices of the n-gon does not leave the interior of the n-gon.)

Exercise 4.6.9 Use strong induction to show that for dividing an $n \times m$ chocolate bar into nm pieces, one has to break it at most $nm - 1$ times.

Exercise 4.6.10 Suppose you have an infinite collection of coins of 2 and 5 Euro cents.

Prove, using strong induction, that you can pay any amount of n Euro cents, where $n \in \mathbb{N}, n \geq 4$.

Give also a proof by assuming the existence of a minimal counter example and reaching a contradiction.

Exercise 4.6.11 Give a recursive definition of the set of all finite directed trees.

Use structural induction to prove that in all finite directed trees the number of edges is one less than the number of vertices.

Exercise 4.6.12 Consider the set $\mathcal{T}$ of binary trees as recursively defined in Example 4.1.7.

A *leaf* of a tree is a vertex with outdegree 0. Denote by l the number of leaves in a tree $T \in \mathcal{T}$. Then $l = (v + 1)/2$ where v is the number of vertices. Prove this using structural induction.

Exercise 4.6.13 Let S be the subset of $\mathbb{Z}$ defined by

$-12, 20 \in S$;

if $x, y \in S$, then $x + y \in S$.

We use structural induction to show that $S = \{4k \mid k \in \mathbb{Z}\}$. The proof is divided into three parts.

a) Show that 4 and -4 are in S.

b) Prove, by structural induction, that $S \subseteq \{4k \mid k \in \mathbb{Z}\}$.

c) Use a) and structural induction to prove that $S \supseteq \{4k \mid k \in \mathbb{Z}\}$.

Exercise 4.6.14 Find in each of the following cases a closed formula for a_n.

1. $a_n = 2a_{n-1} + 2$, $a_1 = 3$;
2. $a_n = a_{n-1} + 3a_{n-2}$, $a_1 = 1$ and $a_2 = 2$;
3. $a_n = a_{n-1} + a_{n-3}$, $a_1 = 1$, $a_2 = 2$, and $a_3 = 0$;
4. $a_n = 2a_{n-1} - a_{n-1}$, $a_1 = 1$ and $a_2 = 0$.

5. Bounding some recurrences

5.1. Growth of a function

Definition 5.1.1 Let f, g be fucntions from $\mathbb{N}$ to $\mathbb{C}$. We write

$$f(n) = O(g(n))$$

and say $f(n)$ is of *order at most* $g(n)$ if there exists a positive constant C_1 such that

$$|f(n)| \leq C_1|g(n)|$$

for all but finitely many $n \in \mathbb{N}$.

We write

$$f(n) = \Omega(g(n))$$

and say $f(n)$ is of *order at most* $g(n)$ if there exists a positive constant C_2 such that

$$|f(n)| \geq C_1|g(n)|$$

for all but finitely many $n \in \mathbb{N}$.

We write

$$f(n) = \Theta(g(n))$$

and say $f(n)$ is of *order* $g(n)$ if $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$.

Example 5.1.2 Let $p(n) = a_0 + a_1 n + \cdots + a_k n^k$, with $a_k \neq 0$ be a polynomial function, then $p(n) = O(n^k)$. Indeed, let $C = |a_0| + |a_1| + \cdots + |a_k|$, then for all $n \in \mathbb{N}$ we have

$$|p(n)| = |a_0 + a_1 n + \cdots + a_k n^k| \leq |a_0| + |a_1|n + \cdots + |a_k|n^k \leq (|a_0| + \cdots + |a_k|)n^k.$$

If all the coefficients a_i are nonnegeative, then $p(n)$ is also $\Omega(n^k)$, as in this case we have for all $n \in \mathbb{N}$ that

$$|p(n)| = a_0 + a_1 n + \cdots + a_k n^k \geq a_k n^k.$$

Example 5.1.3 Consider fucntion $f(n) = \sum_{i=1}^{n} \frac{1}{i}$. Then $f(n) = \Theta(\ln(n))$, where ln denotes the natural logarithm. Indeed, as $\ln(n) = \int_1^n \frac{1}{x} dx$, we find

$$\sum_{i=2}^{n} \frac{1}{i} \leq \ln(n) \leq \sum_{i=1}^{n-1} \frac{1}{i}.$$

So, $\ln(n) \leq f(n) \leq \ln(n) + 1 \leq 2\ln(n)$ for all $n \geq e^2$.

Theorem 5.1.4 *Suppose f, g, h, k are functions from $\mathbb{N}$ to $\mathbb{C}$.*
If $f = O(g)$ and $h = O(k)$, then $f + h = O(|h| + |k|)$ and $fh = O(gk)$.
If $f = \Omega(g)$ and $h = \Omega(k)$, then $|f| + |h| = \Omega(|h| + |k|)$ and $fg = \Omega(gk)$.

Proof. Suppose $f = O(g)$ and $h = O(k)$. Then there exist n_1 and C_1 such that $|f(n)| \leq C_1|g(n)|$ for all $n \geq n_1$ and n_2 and C_2 such that $|h(n)| \leq C_2|k(n)|$ for all $n \geq n_2$. But then for all $n \geq \max(n_1, n_2)$ we have $|f(n) + h(n)| \leq |f(n)| + |h(n)| \leq C_1|g(n)| + C_2|k(n)| \leq (C_1 + C_2)(|g(n)| + |k(n)|)$ and $|f(n)h(n)| = |f(n)| \cdot |h(n)| \leq C_1|g(n)| \cdot C_2|k(n)| \leq (C_1C_2)(|g(n)k(n)|$.

Suppose $f = \Omega(g)$ and $h = \Omega(k)$. Then there exist n_1 and C_1 such that $|f(n)| \geq C_1|g(n)|$ for all $n \geq n_1$ and n_2 and C_2 such that $|h(n)| \geq C_2|k(n)|$ for all $n \geq n_2$. But then for all $n \geq \max(n_1, n_2)$ we have $|f(n)| + |h(n)| \geq C_1|g(n)| + C_2|k(n)| \geq \min(C_1 + C_2)(|g(n)| + |k(n)|)$ and $|f(n)h(n)| = |f(n)| \cdot |h(n)| \geq C_1|g(n)| \cdot C_2|k(n)| = (C_1C_2)(|g(n)k(n)|$.

In the following list we collect a few basic Theta-function with which arbitrary functions are usually compared.

Symbol	Name
$\Theta(1)$	Constant
$\Theta(n)$	linear
$\Theta(n^m)$	Polynomial
$\Theta(m^n)$ with $m > 1$	Exponential
$\Theta(n!)$	Factorial
$\Theta(\ln(n))$	logariithmic
$\Theta(n \ln(n))$	n log n
$\Theta(\ln \ln(n))$	log log n

Example 5.1.5 The function $f : \mathbb{N} \to \mathbb{N}$ given by $f(n) = \ln(n!)$ is $\Theta(n \ln(n))$.
Indeed, $f = O(n \ln(n))$ since

$$f(n) = \ln(n!) = \ln(n) + \ln(n-1) + \cdots + \ln(2) + \ln(1) \leq n \ln(n)$$

for all $n \in \mathbb{N}$. But we also have

$$\begin{aligned} f(n) &= \ln(n) + \ln(n-1) + \cdots + \ln(2) + \ln(1) \\ &\geq \ln(n) + \ln(n-1) + \cdots + \ln(\lceil \tfrac{n}{2} \rceil) \\ &\geq \lceil \tfrac{n}{2} \rceil \ln(\lceil \tfrac{n}{2} \rceil). \end{aligned}$$

(Here $\lceil x \rceil$ denotes the unique integer m with $m - 1 < x \leq m$.)
Since for all $n \geq e^2$ we have

$$\ln(n/2) = \ln(n) - \ln(2) \geq \ln(n) - \ln(n)/2,$$

we have

$$f(n) \geq \frac{n}{2} \ln(\frac{n}{2}) \geq \frac{1}{4} \cdot n \ln(n),$$

which implies $f = \Omega(n \ln(n))$.

5.2. The Master Theorem

The complexity of an algorithm measures the total cost of applying the algorithm in terms of some basic operations (having cost 1).

Suppose an algorithm splits a problem of size n in a subproblems of the same type, but of smaller size n/b. If the splitting into subproblems costs $g(n)$ and if $f(n)$ represents the cost of basic operations to apply the algorithm, then we see that f satisfies the recurrence

$$f(n) = af(n/b) + g(n).$$

This type of recurrence relation is called a *divide and conquer* relation.

Example 5.2.1 Suppose you have to find the word 'mathematics' in a dictionary. You can start at the first page and search word by word till you have found 'mathematics'. Of course, this is not a very efficient way. You probably do something like the following.

Divide the dictinory into two parts and the dicide to which part the word 'mathematics' belongs. Then divide that part in two equal pieces and find the part where the word 'mathematics' is. You repeat this procedure until you have found the word. This so-called *binary search* is an example of divide-and-conquer. Suppose n is the number of words in the part of the dictonary in which you are looking for the word 'mathematics'. If the number of operations 'divide into 2 parts' and 'check in which part the word is' that is still needed to find the word is denoted by $f(n)$, then $f(n) = f(n/2) + 2$ for even n.

Example 5.2.2 Suppose $a = [a_1, a_2, \ldots, a_n]$ is a list of real numbers. We want to locate the maximum (or minumum) of a. If a has length 1 then a_1 is of course the maximum. If the length of a is larger than 1, then we split a into two subsequences of size $\lfloor n/2 \rfloor$ and $\lceil n/2 \rceil$ respectively. Then we proceed by search for the maximum in both parts and set the maximum of a to be the maximum of the two found maxima. Of course, to find the maximum in each subsequence we use the same procedure. The number of comparisons we have to make to find the maximum is denoted by $f(n)$. The functio f staisfies the following recurrence relation: $f(n) = 2f(n) + 2$, when n is even and $f(n) = f((n-1)/2) + f((n+1)/2) + 2$ when n is odd.

Theorem 5.2.3 (Master Theorem) *Let f be a positive and increasing function that satisfies the recurrence relation*

$$f(n) = af(\lfloor n/b \rfloor) + c,$$

where $a \geq 1$, $b > 1$ is an integer and $c \in \mathbb{R}^+$. Then

$$\begin{aligned} f(n) &= O(n^{\log_b a}) && \text{if } a > 1 \\ &= O(\ln n) && \text{if } a = 1 \end{aligned}$$

Proof. First consider the case that $n = b^k$ for some k. Then we can proof by induction on k that

$$f(n) = a^k f(1) + c\sum_{j-0}^{k-1} a^j.$$

If $a = 1$, then $f(n) = f(1) + ck$ and

$$ck \leq f(n) \leq (c + f(1))k.$$

If $a \neq 1$, then $f(n) = a^k f(1) + c(\frac{a^{k+1}-1}{a-1})$ which implies that

$$f(1)a^k \leq f(n) \leq (f(1) + \frac{ca}{a-1})a^k.$$

Now suppose n is arbitrary. Then there is a $k \in \mathbb{N}$ with $b^k \leq n \leq b^{k+1}$. Since f is an increasing function we have

$$f(b^k) \leq f(n) \leq f(b^{k+1}).$$

So, if $a = 1$ then

$$ck \leq f(n) \leq (c + f(1))(k + 1)$$

and, since $k \leq \log_b(n) \leq k + 1$

$$f(n) = \Theta(\log_b(n)) = \Theta(\ln(n)).$$

If $a > 1$, then

$$f(1)a^k \leq f(n) \leq (f(1) + \frac{ca}{a-1})a \cdot a^k.$$

from which we deduce that

$$f(n) = \Theta(n^{\log_b(a)}).$$

□

Example 5.2.4 The number of search operations in a binary search is given by the recursion $f(n) = f(n/2) + 2$, see 5.2.1. Thus one needs $\Theta(\ln(n))$ operations to do such search.

Example 5.2.5 searching a maximum in a sequence as in 5.2.2 requires $\Theta(n^{\log_2(2)}) = \Theta(n)$ comparisons.

5.3. Exercises

Exercise 5.3.1 Select a Theta-notation from the table in Section 5.1. for the function f in each of the following functions.

1. $f(n) = 6n^3 + \ln(n)$;
2. $f(n) = e^n + n^6$;
3. $f(n) = \ln(n^5 + n^4 + n^3)$;
4. $f(n) = \frac{(n+1)(n+2)}{n+3}$;
5. $f(n) = \frac{(n+1)(n+2)}{n+3}$;
6. $f(n) = \frac{n^2+\ln(n)}{n+1}$;

Exercise 5.3.2 Suppose that $f(n) = 2f(n/2) + 3$ when n is even. Find $f(1024)$

Exercise 5.3.3 Find $f(3^k)$ when f satisfies the recursion $f(n) = 2f(n/3) + 4$ and $f(1) = 1$.

Exercise 5.3.4 In a computer addition, subtraction of intgeres and multiplications with powers of 2 (as these are shifts in the binary representation) can be done in $\Theta(n)$ bit-operations, where n is the number of bits needed for the representation of an integer (i.e., the number of digits in the binary representation of the integer). Multiplication is harder.

Suppose a and b are to integers with binary representation

$$a = (a_{2n-1}a_{2n-2}\cdots a_1a_0)_2,$$

$$b = (b_{2n-1}b_{2n-2}\cdots b_1b_0)_2.$$

The product ab can be compute in the following way. Notice that $a = 2^nA_1 + A_0$ and $b = 2^nB_1 + B_0$ where

$$A_1 = (a_{2n-1}a_{2n-2}\cdots a_n)_2,$$
$$A_0 = (a_{n-1}a_{n-2}\cdots a_0)_2,$$
$$B_1 = (b_{2n-1}b_{2n-2}\cdots b_n)_2,$$
$$A_0 = (b_{n-1}b_{n-2}\cdots b_0)_2,$$

and

$$ab = (2^{2n} + 2^n)A_1B_1 + 2^n(A_1 - A_0)(B_0 - B_1) + (2^n + 1)A_0B_0.$$

If $f(n)$ denotes the number of bit operations needed for the multiplication of a and b then $f(n) = 3f(n) + Cn$. Prove this.

6. Graphs

6.1. Graphs

As we noticed before, a directed graph with vertex set A and edge set R is nothing else than a binary relation R on a set A. An ordinary graph with vertex set A has as edges subsets of size two from A. A symmetric and irreflexive relation R is often identified with the ordinary graph with edges $\{a, b\}$ where $(a, b) \in R$.

Many questions about relations can be phrased in terms of graphs. In this section we will discuss some of them. Although many of the definitions and result can also be stated for directed graphs, we will restrict our attention to ordinary graphs. We start with introducing some graph theoretical notation.

Let $\Gamma = (V, E)$ be an ordinary graph with vertex set V and edge set E. A *walk* in Γ is a sequence $(v_1, \ldots, v_n)$ in V such that $\{v_i, v_{i+1}\} \in E$. A *path* in Γ is a sequence $(v_1, \ldots, v_n)$ in V such that $\{v_i, v_{i+1}\} \in E$ and $v_i \neq v_{i+2}$. The *length* of the path $(v_1, \ldots, v_n)$ equals $n - 1$. The point v_1 is the *starting point* of the path and v_n is the *end point*. A *shortest path* from a to b is a path from a to b of minimal length. The *distance* between two points equals the length of a shortest path between them. If there is no such path, the distance is set to infinity. The graph Γ is called *connected* whenever for every two vertices a and b of Γ there is a path with starting point a and end point b. A *cycle* is a path $(v_1, \ldots, v_n)$ with the same starting and end point so, $v_1 = v_n$.

The degree $\deg(v)$ of a vertex $v \in V$ equals the cardinality of the set $\Gamma_v = \{w \in V \mid \{v, w\} \in E\}$. A graph is called *regular* if all its vertices have the same degree, it is called k-regular if all its vertices have degree k.

A connected graph without cycles is called a *tree*. A tree contains vertices of degree 1. These special vertices are called the *leafs* of the tree.

If $\Gamma = (V, E)$ is a graph the the complement $\overline{\Gamma}$ of Γ is the graph with the same vertex set as Γ but with two vertices adjacent if and only if they are not adjacent in Γ. So, if $\binom{V}{2}$ denotes sthe set of all subsets of V of size 2, then $\overline{\Gamma} = (V, \binom{V}{2} \setminus E)$.

6.2. Some special graphs

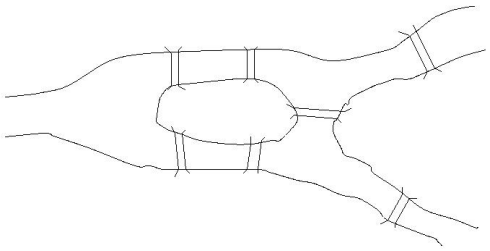
A *complete* graph is a graph in which any two vertices are adjacent. By K_n we denote the complete graph with n vertices.

A graph $\Gamma = (V, E)$ is called *completely bipartite* if its vertex set V can be partitioned into two subsets V_1 and V_2 such that any edge $e \in E$ meets both V_1 and V_2 in a vertex. The complete bipartite graph with $|V_1| = m$ and $|V_2| = n$ will be denoted by $K_{m,n}$.

If $\Gamma = (V, E)$ is a graph, then its *line graph* is the graph whose vertices are the edges of Γ and two of such edges are adjacent if and only if they meet in a vertex of Γ.

6.3. Euler and Hamilton Cycles

Example 6.3.1 One of most famous problems in graph theory is the problem of the bridges of Königsberg (now called Kaliningrad). The Preusian city of Königsberg was divided into 4 parts by a river, one of these parts being an island in the river. The 4 regions of the city were connected by 6 bridges. On sunny Sundays the people of Königsberg used to go walking along the river and over the bridges. The question is, is there a walk using all the bridges once, that brings the pedestrian back to its starting point. The problem was solved by Leonard Euler, who showed that such a walk is not possible.



A cycle in a graph is called an *Euler cycle* if it contains all edges of the graph once.

The following result is due to Euler.

Theorem 6.3.2 *A finite connected graph contains an Euler cycle if and only if all vertices have even degree.*

Proof. Suppose Γ is a graph admitting an Euler cycle E. Suppose v is a vertex of the graph. Then inside the Euler cycle E, we see v just as often as an end point of an edge from E as a starting point. Hence the number of edges on v is even.

Now suppose all vertices of the finite graph Γ have even degree. We start a path of Γ in the vertex v. Each time we arrive in a vertex u, there are only an odd number of edges on u in the path. Except when we arrive back in v. If the path P_1 constructed in this way is not an Euler cycle yet, there is at least one edge, e say, in Γ not visited yet. We may even assume, by connectedness of Γ, that this edge contains a vertex w from P_1. Now we make a path P_2 starting in w containing e. As soon as, in the process of constructing this path, we hit on a vertex of P_1, then, as we only have met an odd number of edges on this vertex, there has to be an edge not in P_1 and not yet part of P_2. So, the path P_2 may be constructed in such a way that it has no edges in common with P_1. The paths P_1 and P_2 can be combined to a path P (with more edges than P_1) in the following way. Start in v, follow the path P_1 until one reaches w, then follow

the path P_2 completely, and finally continue from w over the path P_1 to end in v. Repeating this process will eventually yield an Euler cycle in Γ. □

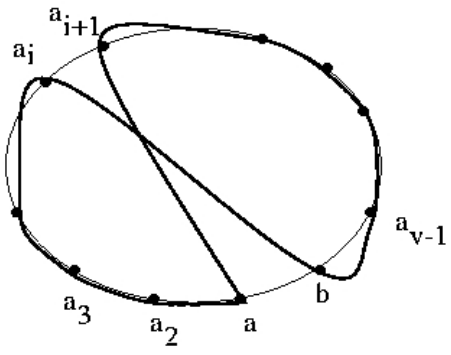
Notice that the above not only proves the theorem, but also describes an algorithm to find an Euler cycle.

A *Hamilton cycle* in a graph Γ is a cycle in the graph containing every vertex exactly ones. It does not seem to be possible to given a characterization of graphs admitting Hamilton cycles, similar to Theorem 6.6.2. However, the following result due to Ore, gives a partial result in this direction.

Theorem 6.3.3 *If in the finite graph Γ on v vertices for every two nonadjacent vertices a and b we have $\deg(a) + \deg(b) \geq v$, then Γ contains a Hamilton cycle.*

Proof. Suppose Γ is a finite graph on v vertices and for any two nonadjacent vertices a and b we have $\deg(a) + \deg(b) \geq v$. Suppose Γ contains no Hamilton cycle. We add edges to Γ as long as possible, but we avoid producing a Hamilton cycle. Since the complete graph on v vertices admits a Hamilton cycle, we end up with a graph Γ_0 in which there is no Hamilton cycle, but addition of any new edges produces one. Notice that, also in Γ_0, we still have that for any two nonadjacent vertices a and b the sum $\deg(a) + \deg(b)$ is greater or equal than v.

Now suppose a, b are nonadjacent vertices. Then after adding the edge $\{a, b\}$ to Γ_0 we obtain a Hamilton cycle $a = a_0, a_1, \ldots, a_v = b$ in Γ_0. Since $\deg(a) + \deg(b) \geq v$, there are two vertices a_i and a_{i+1} with b adjacent to a_i and a adjacent to a_{i+1}. However, then consider the path $a = a_1, \ldots, a_i, b = a_v, a_{v-1}, \ldots, a_{i+1}, a$ in Γ_0. This is a Hamilton cycle and contradicts our assumption on Γ_0. Thus Γ has to have a Hamilton cycle.□



6.4. Spanning Trees and Search Algorithms

Example 6.4.1 A search engine at the university wants to search all the web servers

on campus. These servers are connected by cables within an intranet. To do its search the search engine does not want to put too much load on the connections between the different web servers. In fact, it wants to use as as few connections as possible. If we represent the computer network as a graph with the web servers (and search engine) as vertices, two vertices connected, if and only if there is a cable connecting them, then we can phrase the problem as follows: find a connected subgraph on all the vertices with as few edges as possible.

Trees are graphs with the least number of vertices as follows from the following theorem.

Theorem 6.4.2 *Let Γ be a finite connected graph on v vertices. Then Γ contains at least $v-1$ edges, with equality if and only if Γ is a tree.*

Proof. Notice that in a connected graph each vertex is on at least one edge. We prove this result with induction to the number v of vertices in the graph.

Clearly, for $v = 1$ the result is true. Now suppose $v > 1$ and suppose Γ is a tree. Then removing a leaf of Γ, together with the unique edge on this leaf, yields a tree Γ' with $v-1$ vertices. By induction Γ' contains $v-2$ edges. Thus Γ contains $v-1$ edges.

Now suppose Γ contains $\leq v-1$ edges. Since $|E| \cdot 2 = 2 \cdot (v-1) = \Sigma_{v \in V} \deg(v)$, we find at least one vertex, x say, of degree 1. Removing this vertex and the unique edge on it from Γ yields a connected graph Γ' with $v-1$ vertices and $\leq v-2$ edges. By induction, Γ' is a tree with $v-2$ edges. But then, since the vertex x has degree 1 in Γ, we also find Γ to be a tree with $v-1$ edges. □

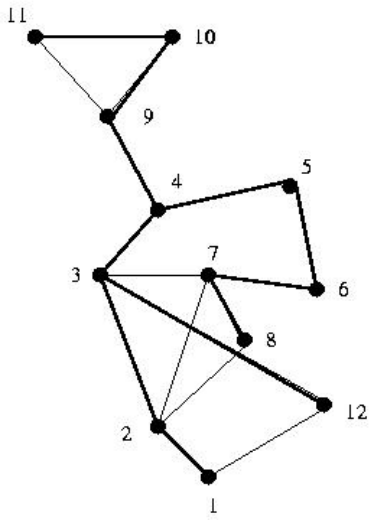
A *spanning tree* of a graph Γ is a subgraph of Γ which is a tree, containing all vertices and some edges of Γ. So the problem of Example 6.4.1 is equivalent to finding a spanning tree in a graph. Now we describe two search algorithms that in fact construct spanning trees.

Algorithm 6.4.3 [Depth First Search and Breadth First Search] Consider a finite connected graph Γ. Fix a vertex v of Γ and label it by $\mathsf{Label} := 1$. The vertex v will be the root of a spanning tree in Γ. We construct this tree now as follows.

While there is a labeled vertex that has a neighbor not in the tree, find the vertex, say w with the highest label that has neighbors not in the tree. Add the edge on w and one of its neighbors not yet in the tree to the tree, and label this neighbor by $\mathsf{Label} := \mathsf{Label} + 1$.

This algorithm is called "Depth First Search". "Breadth First Search" is a similar algorithm. However, here the while loop reads a little bit differently:

While there is a labeled vertex that has an unlabeled neighbor, find the vertex, say w with the smallest label that has neighbors not in the tree. Add the edge on w and one of its neighbors not yet in the tree to the tree, and label this neighbor by $\mathsf{Label} := \mathsf{Label} + 1$.



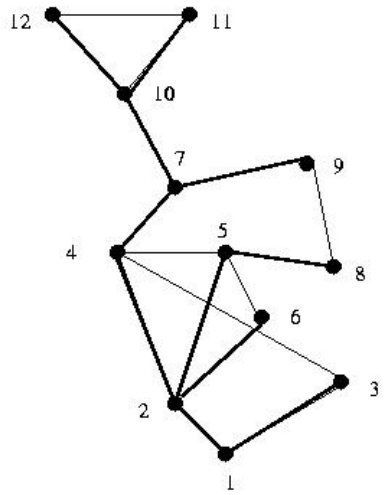
Clearly both algorithms stop after at most $|\Gamma|$ steps and do indeed yield a tree. Actually they both construct a spanning tree. To see this we have to show that every vertex of the graph gets a label. Suppose not, then by connectivity of the graph, there is a path starting with a labeled vertex w and ending with an unlabeled vertex u. walking through this path we will find a labeled vertex adjacent to an unlabeled one. However, this means the while-loop is not finished yet.

Actually the above algorithms imply the following theorem.

Theorem 6.4.4 *Let Γ be a finite connected graph, then Γ contains a spanning tree.*

6.5. Networks

Example 6.5.1 Consider the set A of (big) cities in the Netherlands. As we have seen in 1.5.5, the relation "there is a direct train connection between a and b" defines a binary relation R on this set. The transitive closure of this relation tells us whether we can travel by train from one city to another. It does not tell us what the best or fastest way to travel is. Therefore we need some extra information on the relation. For example, we do not only want to know that there is a train connection from Eindhoven to Tilburg, but we also want to know how long it takes the train to get form Eindhoven to Tilburg. So, for each $(a, b) \in R$ we want to know the time it costs to travel from a to b. Such extra information can be encode by a so-called "cost function". Having this information it is natural to ask for the fastest connection between two cities, i.e., to find the shortest path from a to b.



A graph $\Gamma = (V, E)$ together with a *cost function* cost on the set E of edges, i.e., a map from E to (usually)$\mathbb{R}$ (or any other set) is called a *network*. In such a network the length of a path $(v_1, \ldots, v_n)$ equals to sum $\Sigma_{i=1}^{n-1}\mathsf{cost}(\{v_i, v_{i+1}\})$. The following algorithm due to Dijkstra describes a way to find a shortest path in a network.

Algorithm 6.5.2 [Dijkstra's shortest path algorithm] Suppose $\Gamma = (V, E)$ is a finite connected graph with cost function $\mathsf{cost} : E \to \mathbb{R}^+$ a positive valued cost function. Given two elements s and t of V, a shortest (i.e., of minimal cost) path from s to t can be found as in the following algorithm.

In the process of the algorithm we will keep track of the sets Done and Remainder and some partial maps DefiniteDistance, EstimatedDistance and Predecessor.

Initiate the algorithm by setting $\mathsf{Done} := \{s\}$, $\mathsf{Remainder} := V \setminus \mathsf{Done}$. The distances are set by $\mathsf{DefiniteDistance}(s) := 0$ and $\mathsf{EstimatedDistance}(r) := \infty$ for all $r \in \mathsf{Remainder}$.

While Remainder contains the element t, determine for all elements r in Remainder the $\mathsf{EstimatedDistance}(r)$, being the minimum of $\mathsf{DefiniteDistance}(d) + \mathsf{cost}(d, r)$, where d runs through the set of elements $n \in \mathsf{Done}$ with $\{n, r\} \in E$. Moreover, set $\mathsf{Predecessor}(r)$ to be one of the elements d for which this minimum is attained.

For those elements $r \in \mathsf{Remainder}$ for which $\mathsf{EstimatedDistance}(r)$ is minimal, we can decide that their distance to s is actually equal to $\mathsf{EstimatedDistance}(r)$. So, for those r we set $\mathsf{DefiniteDistance}(r) = \mathsf{EstimatedDistance}(r)$, and we remove these points from Remainder and add them to Done.

Since in each step of the While-loop at least one element is added to Done, the algorithm will terminate and we will find the minimal length $\mathsf{DefiniteDistance}(t)$ of

a path from s to t. Moreover, with the help of Predecessor we even can find a path realizing this minimal length.

Sometimes one is not interested in finding a shortest path in a network, but a longest. Notice that this question only makes sense if we are dealing with a network without cycles, or more in particular, when we are dealing with a *directed network* without cycles. The following variation on Dijkstra's shortest path algorithm yields a solution to that problem.

Algorithm 6.5.3 [Dijkstra's longest path algorithm] First we notice that we may assume that there are no cycles in the network. For otherwise the problem does not make sense.

Again we set Done $:= \{s\}$, Remainder $:= V \setminus$ Done. The distances, however, are set by DefiniteDistance$(s) := 0$ and EstimatedDistance$(r) := 0$ for all $r \in$ Remainder.

While Remainder contains the element t, determine for all elements r in Remainder which can only be reached by an edge starting in Done the DefiniteDistance(r), being the maximum DefiniteDistance$(d)+$cost(d, r), where d runs through the set of elements $n \in$ Done with $\{n, r\} \in E$. Moreover, set Predecessor(r) to be one of the elements d for which this maximum is attained.

We remove these points from Remainder and add them to Done.

Since in each step of the While-loop at least one element is added to Done, then algorithm will terminate, and we will find the maximal length DefiniteDistance(t) of a path from s to t. Moreover, with the help of Predecessor we even can find a path realizing this maximal length.

Algorithm 6.5.4 [Kruskal's Greedy Algorithm] Suppose Γ is a network. To find a minimal spanning tree, we first sort the edges with respect to their weight (or cost). We start with the minimal edge and each time we add an edge of minimal weight to the graph which does not close a cycle. The resulting subgraph will be a spanning tree of minimal weight.

Algorithm 6.5.5 [Prim's Algorithm] In Kruskal's Greedy Algorithm 6.5.4 we first have to sort the edges. The following algorithm avoid this. Start with a vertex v and put it into the set T. At this vertex choose a neighbor outside T on an edge with minimal weight. Add the neighbor and edge to the subgraph T. Now choose a minimal edge among the edges with a vertex in T and one outside T a vertex outside T and add it, together with its end points to T. Repeat this procedure till we have all vertices in T.

Proposition 6.5.6 *Kruskal's Greedy Algorithm and Prim's Algorithm provide us with a minimal spanning tree.*

Proof. Let T_m be the tree obtained after m steps in Kruskal's or Prim's algorithm. With induction on m we prove that there exists a minimal spanning tree T containing T_m. In particular, taking m to be the number of vertices in Γ, we find that the algorithms indeed yield a minimal spanning tree.

For $m = 1$ the graph T_m is just one vertex and hence contained in any minimal spanning tree. Suppose $T_m \subseteq T$ for some minimal spanning tree T. Then in the next step of the algorithm we add an edge $\{v, w\}$ to T_m to obtain T_{m+1}. If $\{v, w\}$ is also an edge of T, then clearly $T_{m+1} \subseteq T$. Thus assume T does not contain $\{v, w\}$. This implies that $\{v, w\}$ with some vertices of T forms a cycle C. Inside that cycle there is an edge $\{v', w'\} \neq \{v, w\}$ with $v' \in T_m$ but $w' \notin T_m$.

By the choice of $\{v, w\}$ in the algorithm, we have $\mathsf{cost}(\{v, w\}) \leq \mathsf{cost}(\{v', w'\}$. Hence, replacing the edge $\{v', w'\}$ in T by $\{v, w\}$ we obtain again a spanning tree, T' say, with $\mathsf{cost}(T) \geq \mathsf{cost}(T')$. In particular, T' is also a minimal spanning tree containing T_{m+1}.

□

6.6. Planar Graphs

A graph is called *planar* if one can draw the graph in the plane (on paper) such that no two edges cross. If a graph is planar, then it is possible to draw it in many different ways. Such a drawing is called a *map* of the graph.

Suppose Γ is a finite planar graph mapped into the plane. Then the following two constructions lead to a new graph also mapped into the plane.

- Add a new vertex and connect it to a vertex of Γ via an edge not crossing any edge of the map.
- Connect two non adjacent vertices of Γ by an edge not crossing any edge of the map.

Actually it is not hard to see that any planar map of a finite graph can be obtained by the appplying the above procedure $|E|$ times, (where E is the edges set of the graph) starting from a graph with a single vertex.

A map of a graph divides the plane in various different *regions*. The degree of a region is the number of edges at its border. As each edge of the graph borders exactly two regions we find the following formula for a finite graph $\Gamma = (V, E)$.

Lemma 6.6.1 *Let $\Gamma = (V, E)$ be a finite planar graph mapped into the plane. Then*

$$\sum_{\mathrm{r}\in\mathrm{R}} \mathrm{degree}(R) = 2|E|,$$

where R is the set of regions of the map of Γ in the plane.

The famous mathematician Euler proved the following formula.

Theorem 6.6.2 [Euler's Formula] $|V| - |E| + |R| = 2$.

Proof. We will prove the theorem using structural induction.

For the graph with just one vertex and no edges the results holds true.

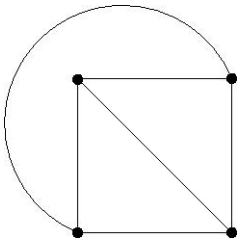
Now any planar map can be obtained from this graph with a single vertex by applying step 1 or 2 described above.

In step 1, no new region is createdm but both V and E get bigger by 1. So $|V| - |E| + |R|$ does not alter.

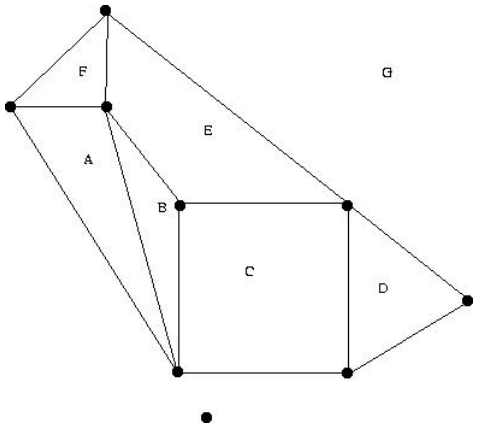
In step 2, $|V|$ remains the same, $|E|$ increases by 1 but also the number of regions increases by 1. Again $|V| - |E| + |R|$ does not alter.

Hence by structural induction, we have proved the theorem. □

Example 6.6.3 A map of the complete graph K_4 in the plane.



A planar graph with 8 vertices, 13 edges and 7 regions.



A consequence of Euler's formula is the following.

Proposition 6.6.4 *Suppose $\Gamma = (V, E)$ is a planar graph with at least 3 vertices. If every region of the graph is bounded by at least e edges, then $2 \leq |V| - (1 - 2/e)|E|$.*

In particular, if Γ is connected and contains at 3 vertices, then $|E| \leq 3|V| - 6$.

Proof. Let R the set of regions of a map of Γ into the plane. Then by 6.6.2 we have

$$|V| - |E| + |R| = 2.$$

As each region is bounded by at least e edges, 6.6.1 we find

$$2|E| \geq e|R|.$$

But then we find

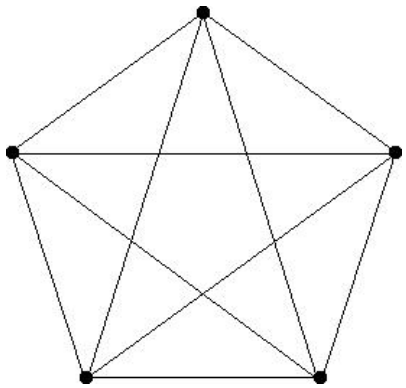
$$2 = |V| - |E| + |R| \leq |V| - (1 - 2/e)|E|,$$

which implies

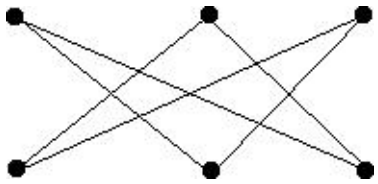
$$|E| + 2 \leq 3|V| - 6.$$

If Γ is connected and contains at 3 vertices, then every region is bounded by at least 3 edges, so we obtain $|E| \leq 3|V| - 6$. □

Example 6.6.5 The graph K_5 is not planar. Indeed, K_5 has 5 vertices and 10 edges, so $10 = |E| > 3|V| - 6 = 9$.



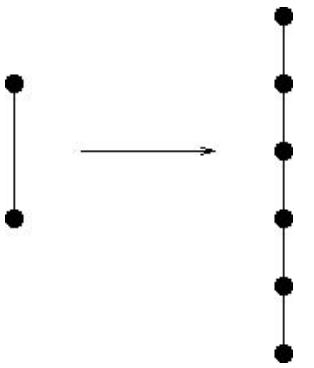
Consider the graph $K_{3,3}$. This graph has 6 vertices and 9 edges, so it does satisfy the conditions as given in the above proposition. However, $K_{3,3}$ is not planar. We will prove this by contradiction. So suppose $K_{3,3}$ is planar. Then by Euler's formula any map of $K_{3,3}$ has 5 regions. Since $K_{3,3}$ does not contain triangles, any region is bounded by at least 4 edges. So, we can apply the above proposition to obtain a contradiction.



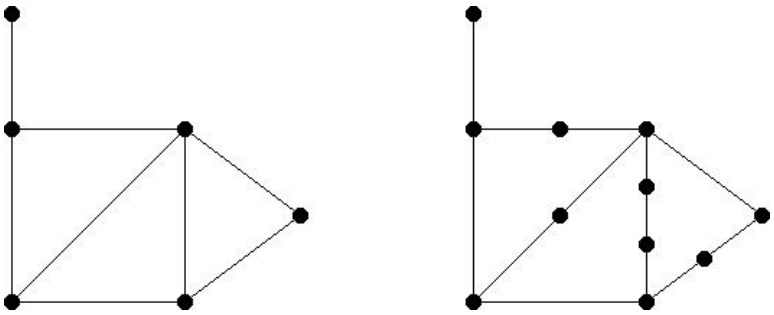
In 1930, the polish mathematician Kuratowski obtained the following beautiful characterization of planar graphs.

Theorem 6.6.6 [Kuratowski's Theorem] *A finite graph* Γ *is planar if and only if it contains no subgraph homeomorphic to* K_5 *or* $K_{3,3}$.

Two graphs are called *homeomorphic* if one can obtain one graph from the other, by replacing edges by strings.



Example 6.6.7 Below you see two homeomorphic graphs.



6.7. Graph Colorings

Consider a graph $\Gamma = (V, E)$. A coloring of Γ is an assigment of colors to the vertices of Γ such that two adjacent vertices have different colors. The minimal number of colors needed to color Γ is called the *chromatic number* of Γ.

One of the most famous results in graph theory is the following theorem due to Appel and Haken.

Theorem 6.7.1 [Four Color Theorem] *Any finite planar graph can be colored with at most* 4 *colors.*

6.8. Exercises

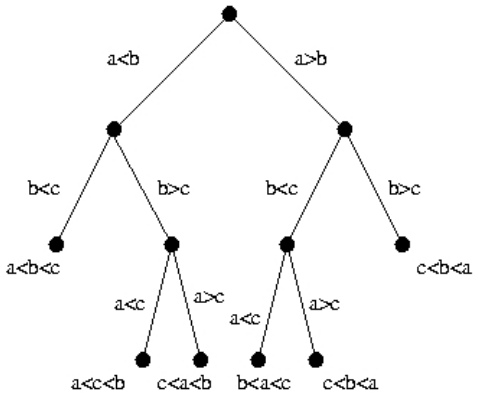
Exercise 6.8.1 A graph Γ is called regular of degree d if all its vertices have degree d. Show that in a finite regular graph $\Gamma = (V, E)$ of degree d we have

$$|V| \cdot d = 2 \cdot |E|.$$

Exercise 6.8.2 A tree Γ is called binary if all its vertices either have degree ≤ 3 or are leafs and one vertex, called the root and denoted by r, is a vertex of degree 2.

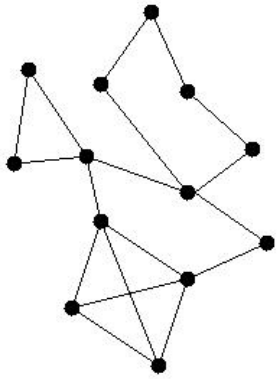
Let Γ be a finite binary tree. Fix a vertex r (called root) of Γ. By l we denote the number of leafs of Γ and by d the maximal distance of a vertex v in Γ to r. Prove that $l \leq 2^d$.

For sorting algorithms based on comparison of two elements we can make a decision tree. For example, for sorting the three elements $\{a, b, c\}$ the decision tree looks as follows:



How many leaves does such a decision tree have when sorting n elements? Suppose r is the starting point of the decision tree. Can you give a lower bound on d? What does this mean for the complexity of sorting algorithms based on comparison of two elements?

Exercise 6.8.3 Find spanning trees in the following graph using both "Depth first search" and "Breadth first search".



Exercise 6.8.4 Suppose T is a spanning tree in a finite connected graph Γ. If e is an edge of Γ which is not in T, then there is an edge d of T, such that replacing d by e in T yields again a spanning tree of Γ. Prove this.

Exercise 6.8.5 How can one use Theorem 6.6.2 to solve the problem of the Königsberger bridges, see 6.3.1?

Exercise 6.8.6 An *Euler path* is a path from a vertex a to a vertex b in a graph containing all edges of the graph just ones. Show that a finite connected graph Γ contains an Euler path from a to b if and only if a and b are the only vertices of odd degree in Γ.

Exercise 6.8.7 How does Theorem 6.6.2 generalize to directed graphs?

Exercise 6.8.8 Can one find an Euler cycle in the cube? And a Hamilton cycle?

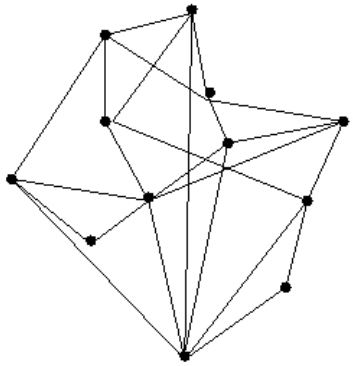
Exercise 6.8.9 Can one find an Euler cycle in the following graph? And a Hamilton cycle?

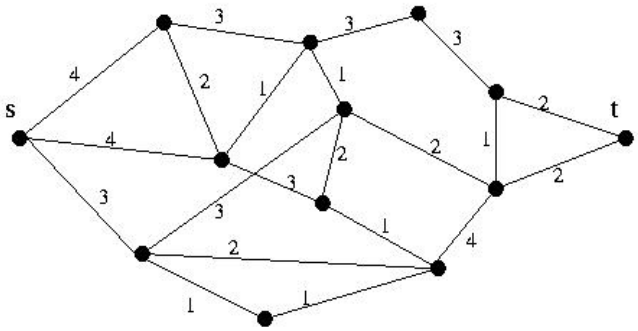
Exercise 6.8.10 Use Dijkstra's algorithm to find both the shortest path from s to t.

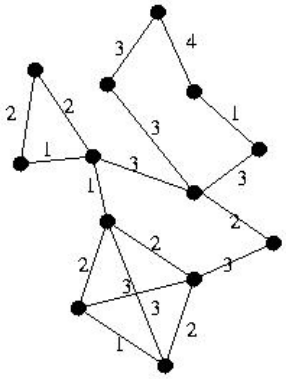
If we assume that one can only walk through the network from left to right, then what is the longest path from s to t?

Exercise 6.8.11 Apply both Kruskal's Greedy Algorithm and Prim's Algorithm to find a minimal spanning tree in the following network.

Exercise 6.8.12 The diameter of a graph is the maximal distance between any of its vertices. Describe an algorithm that determines the diameter of a finite connected graph.




s
t



7. Lattices and Boolean Algebras

7.1. Lattices

Definition 7.1.1 Let $(P, \sqsubseteq)$ be a partially order set and $A \subseteq P$ a subset of P. An element $p \in P$ is called an *upper bound* for A if $a \sqsubseteq p$ for all $a \in A$. It is called a *lower bound* for A if $p \sqsubseteq a$ for all $a \in A$. If the set of all upper bounds of A has a smallest element, then this element is called the *supremum* or *least upper bound* of A. If it exists, it is denoted by $\sup A$.

Similarly the largest lower bound of A (if it exists) is called the *infimum* or *greatest lower bound* of A and is denoted by $\inf A$.

Example 7.1.2

- If S is a set and $P = \mathcal{P}(S)$ the poset of all subsets of S with relation $\subseteq$, then for any subset X of P the supremum $\sup X$ equals $\bigcup_{x \in X} x$ and the infimum $\inf X$ equals $\bigcap_{x \in X} x$.
- Consider the set $\mathbb{N}$ of natural numbers with order relation $|$, "is a divisor of". Then the supremum of two elements $a, b \in \mathbb{N}$ equals $\operatorname{lcm}(a, b)$. The greatest common divisor $\gcd(a, b)$ is the infimum of $\{a, b\}$.
- Suppose X and Y are two sets and P is the set of all partial maps from X to Y. We consider the order $\sqsubseteq$ on P as in Example **??**. If f and g are two distinct maps from X to Y, then there is no supremum of f and g, as there is no partial map having the same graph as both f and g. But the infimum of f and g is the partial map whose graph is the intersection of the graph of f and g.

Definition 7.1.3 A poset $(P, \sqsubseteq)$ is called a *lattice*, if for all $x, y \in P$ the subset $\{x, y\}$ of P has a supremum and an infimum. The supremum of x and y is denoted by $x \sqcup y$ and the infimum as $x \sqcap y$.

Example 7.1.4 Here are some examples of lattices we already encountered before.

- $(\mathbb{R}, \leq)$ is a lattice. If $x, y \in \mathbb{R}$, then $\sup\{x, y\} = \max\{x, y\}$ and $\inf\{x, y\} = \min\{x, y\}$.
- If S is a set and $P = \mathcal{P}(S)$ the poset of all subsets of S with relation $\subseteq$, then P is a lattice with $\sqcap = \cap$ and $\sqcup = \cup$.
- The poset $(\mathbb{N}, |)$ of natural numbers with order relation $|$ is a lattice with the least common multiple as $\sqcup$ and the greatest common divisor as $\sqcap$.

Theorem 7.1.5 *Let $(L, \sqsubseteq)$ be a lattice. Then for all $x, y, z \in L$ we have the following:*

1. $x \sqcup x = x$ *and* $x \sqcap x = x$;
2. $x \sqcup y = y \sqcup x$ *and* $x \sqcap y = y \sqcap x$;

3. $x \sqcup (y \sqcup z) = (x \sqcup y) \sqcup z$ *and* $x \sqcap (y \sqcap z) = (x \sqcap y) \sqcap z$*;*

4. $x \sqcup (x \sqcap y) = x \sqcap (x \sqcup y) = x$*.*

Proof. The first two statements are obvious. To prove the third, we will show that $x \sqcup (y \sqcup z)$ is the supremum of the set $\{x, y, z\}$. Clearly, $x \sqcup (y \sqcup z)$ is an upper bound for x and $y \sqcup z$ and thus also for y and z. Any other upper bound u of x, y and z, is also an upper bound of x and $y \sqcup z$ and therefore an upper bound for $x \sqcup (y \sqcup z)$. This implies that $x \sqcup (y \sqcup z)$ is the smallest upper bound of x, y and z. But so is also $(x \sqcup y) \sqcup z$ and we obtain the first statement of (iii). The second equation follows from the first by considering the dual poset.

It remains to prove (iv). Clearly x is an upper bound of x and $x \sqcap y$. For any other upper bound u of x and $x \sqcap y$ we clearly have $x \sqsubseteq u$. Thus x equals the smallest common upper bound $x \sqcup (x \sqcap y)$ of x and $x \sqcap y$.

Dually we obtain $x \sqcap (x \sqcup y) = x$. □

Remark 7.1.6 Actually, if L is a set with two operations $\sqcup$ and $\sqcap$ from $L \times L \to L$ satisfying (i) upto (iv) of the previous Theorem, then the relation $\sqsubseteq$ given by $x \sqsubseteq y$ if and only if $x = x \sqcap y$ defines a lattice structure on L.

The standard example of a lattice is the lattice of all subsets of a set V. All four properties listed above can easily be checked in this particular example. However not all laws carry over. The so-called distributive laws

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

and

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

for all $A, B, C \subset V$, do not hold in general. A weaker version, however, is valid:

Theorem 7.1.7 *Let* $(L, \sqsubseteq)$ *be a lattice. Then for all* $x, y, z \in L$ *we have the following:*

$$x \sqcup (y \sqcap z) \sqsubseteq (x \sqcup y) \sqcap (x \sqcup z)$$

and dually,

$$x \sqcap (y \sqcup z) \sqsupseteq (x \sqcap y) \sqcup (x \sqcap z).$$

Proof. Let $x, y, z \in L$. Then we have that $x \sqcup y$ is an upper bound for x and for y. Furthermore, as $y \sqcap z \sqsubseteq y$, we find that y is an upper bound for $y \sqcap z$. Thus $x \sqcup y$ is an upper bound for both x and $y \sqcap z$. Hence we have $x \sqcup (y \sqcap z) \sqsubseteq x \sqcup y$. Similarly we find that $x \sqcup (y \sqcap z) \sqsubseteq x \sqcup z$. But then we find $x \sqcup (y \sqcap z) \sqsubseteq (x \sqcup y) \sqcap (x \sqcup z)$.

The second statement is just the dual of the first. □

Definition 7.1.8 Lattices in which we have for all x, y, z

$$x \sqcup (y \sqcap z) = (x \sqcup y) \sqcap (x \sqcup z)$$

and dually,

$$x \sqcap (y \sqcup z) = (x \sqcap y) \sqcup (x \sqcap z)$$

are called *distributive* laticces.

Example 7.1.9 Consider the poset $(P(V), \subseteq)$ of all linear subspaces of a vector space V. Let X, Y, Z be three distinct 1-dimensional subspaces inside a 2-dimensional subspace W. Then $X \sqcup (Y \sqcap Z)$ equals X whereas $(X \sqcup Y) \sqcap (X \sqcup Z) = W \sqcap W = W$. So, we have encountered a nondistributive lattice. (See Exercise 7.5.4.)

Definition 7.1.10 A partial order in which every chain $p_1 \sqsubseteq p_2 \sqsubseteq \dots$ has a supremum is called a *complete partial order* (CPO).

A lattice in which every subset has a supremum and infimum, is called a *complete* lattice. Notice that a complete lattice has a maximal element.

Example 7.1.11 Every finite poset or lattice is complete.

Example 7.1.12 Consider the poset $(\mathcal{P}(V), \subseteq)$ of all subsets of a set V. Then this poset is a lattice. It is also complete. Indeed, for any subset $\mathcal{C}$ of $\mathcal{P}(V)$ the supremum is the union $\bigcup_{c \in \mathcal{C}} c$.

Example 7.1.13 Consider the poset of partial functions from a set A to a set B. This poset is complete. Indeed, if we have a chain $f_1 \sqsubseteq f_2 \sqsubseteq \dots$, then this is a chain of subsets of $A \times B$ with respect to inclusion. Hence $f = \bigcup_{i \in \mathbb{N}} f_i$ is the supremum of the chain.

Example 7.1.14 The poset $(\mathbb{R}, \leq)$ is a lattice, but it is not complete. However, every interval $[a, b]$ with $a < b$ is a complete lattice.

Theorem 7.1.15 *Let $(P, \sqsubseteq)$ be a poset in which every subset has an infimum. Then every subset has also a supremum. In particular, $(P, \sqsubseteq)$ is a complete lattice.*

Proof. Let A be a subset of P. By B we denote the set of all upper bounds of A. The infimum b of B is an upper bound of A. Indeed, every element a is a lower bound for all the elements in B and thus also for the infimum of B. In particular, we find $b \in B$. The element b is the supremum of A. □

7.2. Boolean Algebras

In the previous section we have encountered the operations $\sqcap$ and $\sqcup$ in a lattice $(L, \sqsubseteq)$. In this section we will have a closer look at these operations.

We start with some examples.

Example 7.2.1 Let Ω be a set and $\mathcal{P}(\Omega)$ the set of all subsets of Ω. Then the intersection $\cap$ and the union $\cup$ are two binary operations on $\mathcal{P}(\Omega)$. These operations $\cap$ and $\cup$ satisfy the following laws, where X, Y and Z are elements from $\mathcal{P}(\Omega)$:

- commutativity: $X \cap Y = Y \cap X$ en $X \cup Y = Y \cup X$;
- associativity: $X \cap (Y \cap Z) = (X \cap Y) \cap Z$ en $X \cup (Y \cup Z) = (X \cup Y) \cup Z$;
- absorption: $X \cup (X \cap Y) = X$ en $X \cap (X \cup Y) = X$;
- distributivity: $X \cup (Y \cap Z) = (X \cup Y) \cap (X \cup Z)$ and $X \cap (Y \cup Z) = (X \cap Y) \cup (X \cap Z)$.

The empty set $\emptyset$ and complete set Ω are special elements with respect to these operations $\cap$ en $\cup$. For all $X \in \mathcal{P}(\Omega)$ we have:

- $X \cap \emptyset = \emptyset$;
- $X \cup \Omega = \Omega$.

Finally, for each element X from $\mathcal{P}(\Omega)$ there exists a *complement* X^c with:

- $X \cap X^c = \emptyset$ and $X \cup X^c = \Omega$.

Example 7.2.2 For $n \in \mathbb{N}$ denote by $D(n)$ the set of positive divisors of n. Suppose n is a product of distinct primes.

On $D(n)$ we can consider the operations gcd and lcm. These operations are commutative, associative en distributive. Also the absorption laws hold, for $\mathrm{lcm}(x, \gcd(x, y)) = x$ and $\gcd(x, \mathrm{lcm}(x, y)) = x$. Special elements for these operations are 1 for gcd and n for lcm: $\gcd(1, x) = 1$ en $\mathrm{lcm}(x, n) = n$. Finally, to each $x \in D(n)$ we can associate a complement: n/x. For this complement we have: $\gcd(x, n/x) = 1$ and $\mathrm{lcm}(x, n/x) = n$. Thus $D(n)$ together with the operations gcd, lcm, 1, n and $x \mapsto n/x$ satisfies the same 7 laws as in the previous example.

Definition 7.2.3 *A set V together with the binary operations $\sqcap$ en $\sqcup$, special elements $\bot$ and $\top$, called the* bottom *and* top*, respectively, and a unary operation $X \mapsto X^c$ (the* complement *of v) satisfying for all $X, Y \in V$:*

- *$\sqcap$ and $\sqcup$ are commutative, associative and distributive;*
- *the* absorption laws*: $X \sqcup (X \sqcap Y) = X$ and $X \sqcap (X \sqcup Y) = X$;*

- $X \sqcap \bot = \bot$ *en* $X \sqcup \top = \top$*;*
- $X \sqcup X^c = \top$ *en* $X \sqcap X^c = \bot$*,*

is called a Boolean algebra.

Remark 7.2.4 Boolean algebras are named after the British mathematician George Boole (1815–1864), who was the first to investigate the algebraic laws for the operations $\cup$ en $\cap$ on sets.

Example 7.2.5 Let V be the set of sequences of zeros and ones of length n. On V we define an addition min and multiplication max as follows:

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

$$(a_1, \ldots, a_n)\max(b_1, \ldots, b_n) = (\max(a_1, b_1), \ldots, \max(a_n, b_n))$$

As special elements we have $\bot = (0, \ldots, 0)$ and $\top = (1, \ldots, 1)$. For each $v \in V$ the element v^c is defined as as the unique element with at coordinate i a 0, if and only if the coordinate i of v has value 1. These operation and elements turn V into a Boolean algebra.

The above example and Example 7.2.1 do not really differ. For, if Ω is a set of n elements, then we can fix and order $\omega_1, \ldots, \omega_n$ on these elements of Ω, and every subset X of Ω can be identified with the sequence of zeros and ones with a one at position i if and only if $\omega_i \in X$.

Addition min corresponds to taking intersections, and multiplication max with taking unions.

Example 7.2.6 Suppose Ω is an infinite set and B the Boolean algebra of subsets of Ω as described in 7.2.1. By F we denote the set of subsets of Ω with finitely many points or missing finitely many points form Ω. Then F contains the empty set $\emptyset$ as well as the full set Ω. Moreover, F is closed under $\cup$, $\cap$ and ^{c}. For, if $X, Y \in F$ then also $X \cup Y$, $X \cap Y$ and X^c are in F. (Prove this!) The set F together with the elements $\emptyset$ and Ω, and the operations $\cup$, $\cap$ and ^{c} is also a Boolean algebra.

Lemma 7.2.7 *Let $(V, \sqcap, \sqcup, \bot, \top, v \mapsto v^c)$ be a Boolean algebra. Then V contains a unique bottom and a unique top element. Moreover, for each $v \in V$ the complement v^c is the unique element w in V satisfying $v \cup w = \top$ and $v \cap w = \bot$.*

Proof. Let v be an element in V with $v \sqcap w = v$ for all $w \in V$. Then we have $v = v \sqcap \bot = \bot$. Thus the bottom element $\bot$ is unique.

If v is an element in V with $v \sqcup w = v$ for all $w \in V$. Then we have $v = v \sqcup \top = \top$. Thus also the top element $\top$ is unique.

To prove uniqueness of the complement we first notice that for all $v \in V$ we have:

$$v \sqcup \bot = v \sqcup (v \sqcap \bot) = v, \quad v \sqcap \top = v \sqcap (v \sqcup \top) = v.$$

Suppose, to prove uniqueness of the complement of v, that $v \sqcap w = \bot$ and $v \sqcup w = \top$ for some element $w \in V$. Then the above implies:

$$\begin{aligned} w &= w \sqcap \top \\ &= w \sqcap (v \sqcup v^c) \\ &= (w \sqcap v) \sqcup (w \cap v^c) \\ &= \bot \sqcup (w \sqcap v^c) \\ &= (v \sqcap v^c) \sqcup (w \sqcap v^c) \\ &= (v \sqcup w) \sqcap v^c \\ &= \top \sqcap v^c \\ &= v^c, \end{aligned}$$

and thus $w = v^c$. □

Exercise 7.2.1 Show that in a Boolean algebra we have:

$$\bot^c = \top \quad \text{and} \quad \top^c = \bot .$$

Exercise 7.2.2 Suppose V with the operations $\sqcap, \sqcup, \bot, \top$ and $v \mapsto v^c$ is a Boolean algebra. Show that for all v and w in V we have :

1. $(v^c)^c = v$;
2. $(v \sqcap w)^c = v^c \sqcup w^c$;
3. $(v \sqcup w)^c = v^c \sqcap w^c$.

The last two rules are called the rules of De Morgan.

Exercise 7.2.3 Let $n \in \mathbb{N}$ and denote by $D(n)$ the set of divisors of n. On $D(n)$ we consider the operations as defined in Example 7.2.2. Does this always turn $D(n)$ into a Boolean algebra? Give necessary an sufficient conditions on n to turn $D(n)$ into a Boolean algebra.

Exercise 7.2.4 Prove that there is no Boolean algebra on three elements.

7.3. Examples from Logic and Electrical Engineering

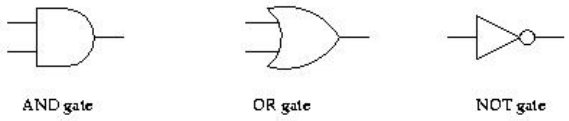
Example 7.3.1 In logic we often make use of the following truth table:

X	Y	X∧Y	X∨Y	¬X
True	True	True	True	False
True	False	False	True	False
False	True	False	True	True
False	False	False	False	True

the symbols X and Y can be interpreted as maps on a (not further specified) set taking the values $True$ or $False$. The first two columns give the various values X and Y can take.

From this table we easily check that $\wedge$, $\vee$ and $\neg$ define a Boolean algebra on the set $\{False, True\}$.

Example 7.3.2 In the design of electrical circuits one also often uses three basic gates, a NOT-gate, an AND-gate and an OR-gate, see [**?**].

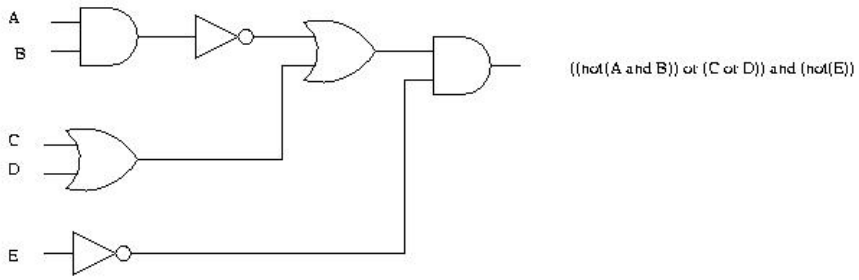


The NOT-gate gives a signal if there is no input signal, and gives no signal if there is an input signal. The AND-gate accepts two signal and returns only a signal if there are indeed two signals coming in. The OR-gate accepts two signal and returns a signal if there is at least one signal coming in. Several of these gates are set together to form the electrical circuit. In such a circuit we have n input channels and one output channel. We denote a signal at a channel by 1 and no signal by 0. The input is then a sequence of n zeros or ones. The output is just 1 or 0. We regard such a circuit as a map from the n input channels to the output channel, or from the set of $0, 1$-sequences of length n to $\{0, 1\}$.

For example, if we consider the a circuit with five input channels A, B, C, D and E as in the figure below, then we can think of A to represent the function which is projection on the A^{th}-coordinate, B is projection on the B^{th}-coordinate, etc. The AND-gate corresponds to taking the minimum, the OR-gate to taking the maximum. So A and B is minimum of the functions A and B, and A or B is the maximum of the functions A and B . The NOT-gate changes the value of a map from 1 to 0 or from 0 to 1.

The circuit displayed in the figure corresponds to the map

$$(\text{not } (A \text{ and } B) \text{ or } (C \text{ or } D)) \text{ and } (\text{not } E).$$

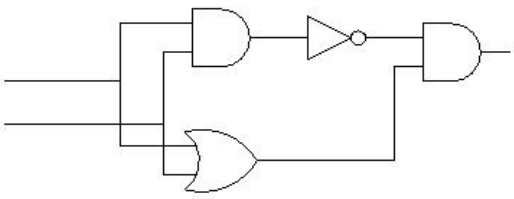


The AND-gate, OR-gate and NOT-gate turn the set of electrical circuits into a Boolean algebra.

Exercise 7.3.1 What is $\bot$ and what is $\top$ in the Boolean algebra of Example 7.3.1? The map $X \wedge Y$ evaluates only in one case to $True$. Construct three other maps that also evaluate only ones to $True$.

Exercise 7.3.2 What is $\bot$ and what is $\top$ in the Boolean algebra of Example 7.3.2? Can you prove that the AND-gate, OR-gate and NOT-gate turn the set of electrical circuits indeed into a Boolean algebra?

Exercise 7.3.3 The following electrical circuit represents the so-called Hotel switch. What does it do? Why is it called Hotel switch? Can you find another element in the



Boolean algebra of electrical circuits representing this Hotel switch?

7.4. The Structure of Boolean Algebras

Proposition 7.4.1 *Let $B = (V, \sqcap, \sqcup, \bot, \top, v \mapsto v^c)$ be a Boolean algebra. For $x, y \in V$ the following three statements are equivalent.*

1. $x \sqcap y = x$;
2. $x \sqcup y = y$;

3. $\exists z \in V : \ y = x \sqcup z$.

Proof. (i)⇒(iii). Suppose (i). Let $z = y$. Then

$$x \sqcup z = (y \sqcap x) \sqcup (y \sqcap \top) = y \sqcap (x \sqcup \top) = y \sqcap \top = y,$$

which proves (iii).

(iii)⇒(ii). Suppose (iii) . Then with z as in (iii):

$$y \sqcup x = x \sqcup z \sqcup x = x \sqcup z = y.$$

Hence (ii).

(ii)⇒(i). If (ii) holds, then, by the second absorption law,

$$y \sqcap x = (y \cup x) \sqcap x = x,$$

from which we deduce (i). □

Definition 7.4.2 Let $B = (V, \sqcap, \sqcup, \bot, \top, v \mapsto v^c)$ be a Boolean algebra. Define the relation $\sqsubseteq$ on V by

$$x \sqsubseteq y \Leftrightarrow x \sqcap y = x$$

for $x, y \in V$. (One of the three equivalent statements from 7.4.1.)

Lemma 7.4.3 *Let $B = (V, \sqcap, \sqcup, \bot, \top, v \mapsto v^c)$ be a Boolean algebra. Then $\sqsubseteq$ defines a partial order on V with minimum $\bot$ and maximum $\top$.*

Proof. Let $x, y, z \in V$.

$\sqsubseteq$ is reflexief: Indeed, since $x = x \sqcap x$, we have $x \sqsubseteq x$.

$\sqsubseteq$ is antisymmetric: Suppose $x \sqsubseteq y \sqsubseteq x$. Then $x = x \sqcap y = y$, so $x = y$.

$\sqsubseteq$ is transitive: Suppose $x \sqsubseteq y \sqsubseteq z$. Then we have $x = y \sqcap x = (z \sqcap y) \sqcap x = z \sqcap (y \sqcap x) = z \sqcap x$, and hence $x \sqsubseteq z$.

We have proved $\sqsubseteq$ to be reflexive, antisymmetric and transitive. Hence it is a partial order.

Finally, since $\bot = \bot \sqcap x$ en $\top = \top \sqcup x$ (see 7.4.1), we find $\bot$ and $\top$ to be the minimum and maximum of $(V, \sqsubseteq)$. □

Definition 7.4.4 An element x of a Boolean algebra is called an *atom* if is there is no other element y than $\bot$ and x itself with $y \sqsubseteq x$.

By Proposition 7.4.1 an element x different from $\bot$ in a Boolean algebra is an atom if and only if for all $y \neq \bot$ we have: if $x \sqcap y = y$, then $y = x$. If $a_1, \ldots, a_m$ are elements in a Boolean algebra, then

$$a_1 \sqcup \cdots \sqcup a_m$$

is well defined ($\sqcup$ is associative). Moreover, this expression is independent of the order of the a_i ($\sqcup$ is commutative) and finally it is independent of the multiplicity of the a_i (for, $a \sqcup a = a$). Therefore, we can express this the element $a_1 \sqcup \cdots \sqcup a_m$ also by

$$\bigsqcup_{1 \leq i \leq m} a_i.$$

Similarly, also

$$a_1 \sqcap \cdots \sqcap a_m$$

is well defined, and we write it as

$$1 \leq i \leq m a_i.$$

Theorem 7.4.5 [Representation Theorem of Finite Boolean Algebras] *Let $B = (V, \sqcap, \sqcup, \bot, \top, v \mapsto v^c)$ be a finite Boolean algebra. Suppose A is the set of atoms in B. Then every element $b \in V$ can be written as*

$$b = \bigsqcup_{a \in A, a \sqsubseteq b} a.$$

Moreover, this expression is unique, i.e., if $b = \bigsqcup_{a \in A'} a$ for some subset A' of A, then $A' = \{a \in A, a \sqsubseteq b\}$.

Proof. Suppose the right hand side of

$$b = \bigsqcup_{a \in A, a \sqsubseteq b} a$$

equals w. Then by 7.5.11 we have $w \sqsubseteq b$. Therefore

$$b = (b \sqcap w) \sqcup (b \sqcap w^c) = w \sqcup (b \sqcap w^c).$$

We will show $b \sqcap w^c = \bot$. For, then we have $b = w \sqcup \bot = w$, and we are done.

If $\alpha \in A$ satisfies $\alpha \sqsubseteq b \sqcap w^c$, then

$$\alpha \sqsubseteq b \quad \text{and} \quad \alpha \sqsubseteq_{a \in A, a \sqsubseteq b} a^c.$$

In particular, $\alpha \sqsubseteq \alpha^c$, so $\alpha = \alpha \sqcap \alpha^c = \bot$. A contradiction. This implies that no element from A is less than or equal to $b \sqcap w^c$. Since V is finite, there is for each element $v \neq \bot$ an atom a with $a \sqsubseteq v$: if v is not an atom, then there is an element $v_1 \neq \bot$ with $v_1 \sqsubseteq v$.

Then v_1 is an atom, or there is an element $v_2 \neq \bot$ with $v_2 \sqsubseteq v_1$. Etc. Conclusion: $b \sqcap w^c = 0$.

Remains to prove the uniqueness of this expression for b. Suppose that $b = \bigsqcup_{a \in A'} a$ for some subset A' of A. Then clearly $A' \subseteq \{a \in A, a \sqsubseteq b\}$. If $a_0 \in \{a \in A \mid a \sqsubseteq b\} \setminus A'$, then

$$a_0 = a_0 \sqcap b = a_0 \sqcap (\bigsqcup_{a \in A'} a) = \bigsqcup_{a \in A'} (a_0 \sqcap a) = \bigsqcup_{a \in A'} \bot = \bot .$$

A contradiction. □

The above theorem states that a finite Boolean algebra B can be identified with the Boolean algebra of subsets of A, the set of atoms of B. In the next section we will make this statement more precise.

Notice that the above result is not true for infinite Boolean algebras. Indeed, the Example 7.2.6 shows that an infinite Boolean algebra can not always be identified with the set of subsets of its atoms.

Example 7.4.6 Consider the Boolean algebra of electrical circuits with n input channels as described in Example 7.3.2. Then the AND-gate corresponds to $\sqcap$ and the OR-gate to $\sqcup$. In terms of maps from the set of $0,1$-sequences of length n to $\{0,1\}$, AND corresponds to taking the minimum of two functions. Hence, the map which sends everything to 0 is the minimal element $\bot$ in this Boolean algebra. The atoms are the elements that take the value 1 only ones. Hence there are just as many atoms as there $0,1$-sequences and that number is 2^n. Now Theorem 7.4.5 implies that there are exactly 2^{2^n} different elements in this Boolean algebra. But as there are also exactly 2^{2^n} different maps from the set of $0,1$-sequences to $\{0,1\}$, this implies that each possible map can be realized by a circuit.

It remains the problem to find for a given map a circuit that realizes the map. Or even better, a small circuit doing so. This is still a difficult task. But our translation of the problem into a problem of Boolean algebras makes it easier to solve this question with the help of the computational power of computers.

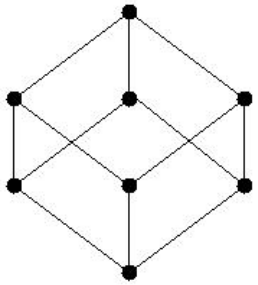
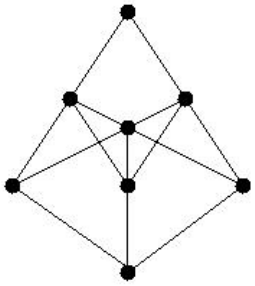
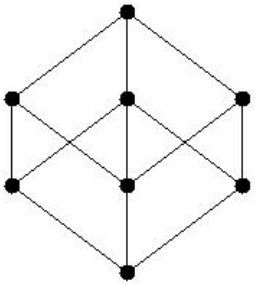
7.5. Exercises

Exercise 7.5.1 Let $(P, \sqsubseteq)$ be a poset and A a subset of P. Prove that an element $a \in A$ is maximal if and only if for all $x \in A$ we have $a \sqsubseteq x$ implies $a = x$.

Exercise 7.5.2 Let $(L, \sqsubseteq)$ be a lattice and $x, y, z \in L$. Prove that

1. $x \sqsubseteq y$ implies $x \sqcup z \sqsubseteq y \sqcup z$.
2. $x \sqsubseteq z$ implies $z \sqcup (x \sqcap y) = z$.

Exercise 7.5.3 In the figure below you see three diagrams. Which of these diagrams are Hasse diagrams? Which of these diagrams represents a lattice?

Exercise 7.5.4 Let V be a vector space. Show that the poset $(P(V), \subseteq)$ is a complete lattice.

Exercise 7.5.5 Consider the poset of partial functions from a set A to a set B as in 7.1.13. This is a complete poset. Prove this.

Exercise 7.5.6 Is the poset $(\{1, 2, 3, 4, \dots\}, |)$ a complete lattice? How about $(\{0, 1, 2, 3, 4, \dots\}, |)$?

Exercise 7.5.7 Suppose $(L, \sqsubseteq)$ is a lattice and $a, b \in L$ with $a \sqsubseteq b$. Prove that $\sqsubseteq$ induces a lattice on the interval $[a, b]$.

Exercise 7.5.8 Let $(L, \sqsubseteq)$ be a lattice. If for all $x, y, z \in L$ we have

$$x \sqcup (y \sqcap z) = (x \sqcup y) \sqcap (x \sqcup z)$$

then we also have for all $x, y, z \in L$ that

$$x \sqcap (y \sqcup z) = (x \sqcap y) \sqcup (x \sqcap z).$$

Prove this. (Hint: use $x \sqcap (y \sqcup z) = (x \sqcap (y \sqcup x)) \sqcap (y \sqcup z)$.)

Exercise 7.5.9 Determine the atoms of all the Boolean algebras from 7.2.2, 7.2.1,7.2.6, and 7.3.1.

Exercise 7.5.10 Prove for elements a, b in a Boolean algebra:

$$a \sqsubseteq b \Leftrightarrow b^c \sqsubseteq a^c.$$

Exercise 7.5.11 Prove, for elements x, y, z in a Boolean algebra,

1. that $x \sqsubseteq z$ and $y \sqsubseteq z$ is equivalent to $x \sqcup y \sqsubseteq z$;
2. that $z \sqsubseteq x$ and $z \sqsubseteq y$ is equivalent to $z \sqsubseteq x \sqcap y$;

3. that $x \sqcap y = \bot$ is equivalent to $x \sqsubseteq y^c$. [hint: look at $x \sqcap (y \sqcup y^c)$.]

Exercise 7.5.12 Let a be an atom in a Boolean algebra and suppose x, y are two arbitrary elements. Prove:

1. $a \sqsubseteq x^c \Leftrightarrow a \not\sqsubseteq x$. [hint: $\Rightarrow$ 7.5.11 (ii); $\Leftarrow$ 7.5.11 (iii).]

2. $a \sqsubseteq x \sqcup y \Leftrightarrow a \sqsubseteq x$ or $a \sqsubseteq y$.

Exercise 7.5.13 Suppose $(V, \cap, \cup, 0, 1, v \mapsto v^c)$ is a Boolean algebra. Show that for all v in V we have:

1. $v \sqcap v = v$;

2. $v \sqcup v = v$.

Exercise 7.5.14 In Exercise 7.2.4 one should prove that there is no Boolean algebra on three elements. With the results of this section at hand, one can prove much more. What order can a finite Boolean algebra have?

References

[1] A. Cohen, H. Cuypers, H. Sterk, *Algebra Interactive!*, Springer Verlag, 1999.