Discrete Structures [*]

Lecture Notes

Steffen van Bakel

Department of Computing, Imperial College London, 180 Queen's Gate, London SW7 2BZ, UK

Autumn, 2017

Introduction

This material is written as notes to accompany a course on Discrete Structures given to first year students in the Department of Computing at Imperial College London.

We introduce the basics of discrete mathematics – in particular, sets, relations, functions, and induction. These notes are intended for beginning students of computer science, who will find that a knowledge of the concepts covered here will aid them in understanding many areas of computer science, like for instance, data types, databases, specification, functional programming, and logic programming.

This course endeavours to give a rigorous account, that forms the underpinnings of many courses in the Computer Science degree. The work is self contained, though rather concise.

Recommended books

- K.H. Rosen. Discrete Mathematics and its Applications, McGraw Hill 1995.
- J.L. Gersting. Mathematical Structures for Computer Science, Freeman 1993.
- J.K. Truss. Discrete Mathematics for Computer Science, Addison-Wesley 1991.
- R. Johnsonbaugh, Discrete Mathematics, 5th ed. Prentice Hall 2000.
- C. Schumacher, Fundamental Notions of Abstract Mathematics, Addison-Wesley, 2001.

Related courses include the mathematical reasoning courses (Logic, Reasoning on Programs, Discrete Mathematics II, Haskell, and Databases I). In particular, we will use some of the notation for formulas introduced in the 'Logic' course:

$$A \wedge B \qquad A \vee B \qquad \neg A \qquad A \Rightarrow B \qquad A \Leftrightarrow B \qquad \forall x\,(A) \qquad \exists x\,(A)$$

as well as the properties of those operations as shown there.

Exercises and sections marked with an asterisk are not part of the examinable material of the course.

[*] These lecture notes are in part based on previous notes by Iain Phillips and Philippa Gardner.

Contents

1 Sets

One of the most fundamental concepts in mathematics is that of a *set*. It is generally seen as 'a collection of things', like 'all even numbers', 'all complex numbers', 'all cars manufactured in England', etc. Although normally we will not be too meticulous on whether some collection is indeed a set, we will make clear that care has to be taken with a formal definition: not every collection is a set.

We will first introduce Georg Cantor's original definition from 1883 together with some notation, much of which was introduced by Giuseppe Peano in 1895.

Definition 1.1 (SETS) *1)* A *set* is a collection of definite and separate objects (or *individuals*) into a whole. Sets are written like

$$\{a_1,\ldots,a_n\} \text{ or } \{x \mid P(x)\}$$

where the former states the members of the set explicitly, and the latter stands for the collection of all objects, ranged over by x, that satisfy the property P (the notation $P(x)$ stands for a formula that contains x); this is called *restriction*, or *comprehension* (see Definition 1.7).

2) The members of a set are also called the *elements* of the set and a set is said to *contain* its elements. We write

$$a \in A$$

when object a is an element of set A and $a \notin A$ when a is not an element of A.

3) We assume that each set A comes equipped with an equality relation '$=_A$', that tells us wether two elements of A are the same; we will drop the subscript on '$=$' when clear from the context.

Notice that $a \in A$ is a statement that is either true or false, so $a \notin A$ is the same as $\neg(a \in A)$, and that it is impossible for $a \in A$ and $a \notin A$ to hold simultaneously. We will often write $a,b \in A$ for $a \in A \wedge b \in A$.

Sets can be finite, i.e. have a finite number of elements (then the number of elements is an element of $I\!N$), or infinite, i.e. have an infinite number of elements (with that number the same as that of the number of elements in $I\!N$, or even larger). Finite sets can be defined by listing their elements. Infinite sets normally are defined using a pattern or restriction.

Here are some examples:

- the two-element set $\mathsf{Bool} = \{\mathsf{True},\mathsf{False}\}$;
- the set of vowels $V = \{a,e,i,o,u\}$;
- an arbitrary set $\{1,2,e,f,5,\mathsf{Imperial}\}$, which is a set containing numbers, letters, and a string of letters with no obvious relationship to each other;
- the set of natural numbers $I\!N = \{0,1,2,3,\ldots\}$, which is read as '$I\!N$ is the set containing the natural numbers 0, 1, 2, 3, etc.';
- the set of integers $\mathbb{Z} = \{\ldots,-3,-2,-1,0,1,2,3,\ldots\}$;
- the set of primes $P = \{p \in I\!N \mid p \textit{ is a prime number}\}$;
- the empty set $\emptyset = \{\,\}$, which contains no elements;
- nested sets, such as the set $\{\{\emptyset\},\{a,e,i,o,u\}\}$ containing the sets $\{\emptyset\}$ and $\{a,e,i,o,u\}$ as its two elements.

The set $I\!N$ is of course an infinite set. The ellipses '$\ldots$' indicate that the remaining elements are given by some rule, which should be apparent from the initial examples. Notice that sets can themselves be members of other sets, as the last example illustrates. There is an important distinction between $\{a,b,c\}$ and $\{\{a,b,c\}\}$ for example: the first set contains three elements, the second only one; or between $\emptyset$ and $\{\emptyset\}$: the latter set is not empty, since it contains a set.

1.1 Russell's paradox

Cantor's set theory, although very natural and intuitive, was shown to be inconsistent by Bertrand Russell in 1901. He showed that we cannot just group any collection and call it a set but need to be more cautious by giving a collection that is defined using Cantor's approach – so should be a set – but cannot exist. The conclusion therefore must be that not every collection of mathematical objects defined this way automatically forms a set. It is often presented as a paradox, but in fact shows that Cantor's definition is badly constructed because it leads to a contradiction (is *inconsistent*).

Theorem 1.2 *The collection* $R \triangleq$ [1] $\{X \text{ is a set} \mid X \notin X\}$ *is not a set.*

Proof: Assume that R is a set. Then either $R \in R$ or $R \notin R$. We analyse the cases:

$(R \in R)$: then by definition of R it follows that $R \notin R$ which is a contradiction.

$(R \notin R)$: then by definition of R it follows that $R \in R$ and again we have a contradiction.

So in both cases we come to a contradiction, hence the initial assumption must be rejected and R cannot be a set. ■

(Notice that it is perfectly possible for a set to be an element of itself. For example, take A to be the set of all sets that contain at least two elements, then clearly A has more than two elements, so $A \in A$; likewise, the set B of all finite sets is itself not finite, so $B \notin B$.)

This counter example led to a more precise definition of the notion of set. In particular, not every collection as defined by Cantor is a set; for example, the collection of all sets is itself not a set. One way to avoid that problem is, when defining a set as $\{x \mid P(x)\}$, to specify from which set the x are taken, and write $\{x \in A \mid P(x)\}$, where A is a set.

We will discuss here the definition of set theory of Ernst Zermelo and Abraham Fränkel from 1908 (see Definition 8.1). Before we can present and discuss that definition, we need to introduce some notions and operations on sets that are used there. For the time being, we will just accept that sets exist, and are often not too worried about checking that something is indeed a set.

1.2 Comparing Sets

We will now define the notion of one set being a subset of (or be contained in) another set, and the related notion of two sets being equal.

Definition 1.3 (SUBSETS) Let B be a set; any collection A of elements of B is a set as well and is called a *subset of* B, written $A \subseteq B$: in other words,

$$A \subseteq B \;\triangleq\; \forall x \in A\ (x \in B)$$

We can write $\forall x\ (x \in A \Rightarrow x \in B)$ for $\forall x \in A\ (x \in B)$; we will use either notation, depending on which is more convenient. Notice that the definition $\forall x\ (x \in A \Rightarrow x \in B)$ talks about all x, although we have not said what kind of object x is. We assume that there exists an 'underlying universe of discourse' when discussing sets: that is, the collection of all possible objects under discussion, which we will assume forms a set.

Example 1.4 Any set is, trivially, a subset of itself. Other simple examples are

$$\begin{aligned} \{a,b\} &\subseteq \{a,b,c\} \\ \mathbb{N} &\subseteq \mathbb{Z} \\ \emptyset &\subseteq \{1,2,5\} \end{aligned}$$

To show that the last example is true we need to show that every element in $\emptyset$ is contained in $\{1,2,5\}$. Since there are no elements in $\emptyset$, this property is vacuously true: therefore $\emptyset$ is a subset of every set.

Proposition 1.5 Let A, B, C be arbitrary sets. If $A \subseteq B$ and $B \subseteq C$, then $A \subseteq C$.

Proof: Assume that A, B, and C are arbitrary sets and that $A \subseteq B$ and $B \subseteq C$. We need to show that, for an arbitrary element x, if x is an element of A then it must also be an element of C.

So assume $x \in A$. By assumption, we know that $A \subseteq B$, and hence by definition of the subset relation also $x \in B$. We also know that $B \subseteq C$, and again by definition of the subset relation also $x \in C$. So assuming that $x \in A$, we have shown that $x \in C$. We have shown this without using any specific knowledge

[1] The symbol '$\triangleq$' can be read as 'is defined as'; in the literature, often the symbol $\Leftrightarrow$ is used, but we will reserve that for statements that are logical consequences of each other, i.e. imply each other through logical reasoning.

about x other than that it is a member of A, so have shown the property for *all* elements of A, so have shown

$$\forall x \in A\ (x \in C)$$

Hence by definition $A \subseteq C$, as required.

Alternatively, more formally, we can write:

$$\begin{array}{rcl} A \subseteq B \wedge B \subseteq C & \triangleq & \forall x \in A\ (x \in B) \wedge \forall x \in B\ (x \in C) \\ & \Rightarrow & \forall x \in A\ (x \in B \wedge \forall y \in B\ (y \in C)) \\ & \Rightarrow & \forall x \in A\ (x \in B \wedge x \in C) \\ & \Rightarrow & \forall x \in A\ (x \in C) \end{array}$$

■

In writing down formal proofs, both approaches are acceptable.

Whenever we introduce a structure (in this case sets), it is important to be able to identify when we consider two such structures to be equal, in other words when they denote the same thing. For sets, two sets are equal when they are subsets of each other, so contain the same elements.

Definition 1.6 (SET EQUALITY) Let A, B be any two sets. Then A and B are equal, written $A = B$, if both $A \subseteq B$ and $B \subseteq A$: that is,

$$A = B \ \triangleq\ A \subseteq B \wedge B \subseteq A$$

So equality between sets is defined through set inclusion, and we can formally only show, for any two sets C and D, that $C = D$ by showing that both $D \subseteq C$ and $C \subseteq D$, even if our intuition 'sees' that C and D are the same.

This definition of equality on sets implies that the order of the elements of a set do not matter: the sets $\{a,b,c\}$ and $\{b,c,a\}$ are equal; mark that we can argue that they are not the same description of a set.[2]

1.3 Constructing Sets

There are several ways of describing sets. So far, we have informally introduced two approaches: either we just list the elements inside curly brackets:

$$V = \{a,e,i,o,u\},\ I\!N = \{0,1,2,\ldots\},\ \{\emptyset,\{a\},\{b\},\{a,b\}\};$$

or we define a set by stating the property that its elements must satisfy:

$$P \ = \ \{p \in I\!N \mid p \textit{ is a prime number}\}$$

The second approach can be generalised to a general axiom of *comprehension*, which states that the collection of elements from a set that satisfy a certain logical property forms a set.

Definition 1.7 (COMPREHENSION) We can construct a (new) set by taking all elements of a set A that satisfies some property $P(x)$, as in

$$\{x \in A \mid P(x)\}$$

where P is a *predicate* (a formula expressed using logical notation).

Example 1.8 Assume $B = \{x \in A \mid P(x)\}$, and $C = \{x \in A \mid Q(x)\}$ and that $\forall x \in A\ (P(x) \Leftrightarrow Q(x))$; if we want to show that $B = C$, we would have to reason as follows:

$$\begin{array}{l} y \in B \ \Rightarrow\ P(y) \ \Rightarrow\ Q(y) \ \Rightarrow\ y \in C \\ y \in C \ \Rightarrow\ Q(y) \ \Rightarrow\ P(y) \ \Rightarrow\ y \in B \end{array}$$

However, it is in general acceptable to write

$$\{x \in A \mid P(x)\} \ = \ \{x \in A \mid Q(x)\}$$

when the property $\forall x \in A\ (P(x) \Leftrightarrow Q(x))$ is evident.

[2] This is comparable to saying that $3+4 = 7$; $3+4$ and 7 are not the same expression, but we consider them to represent the same value.

For set comprehension to be correct, we need that A is known to be a set. This implies that a definition like

$$\mathit{IR} \;=\; \{x \mid x \text{ is a real number}\}$$

would not formally be acceptable: first of all, it 'depends on itself', but more importantly, we have not specified from what set x is selected; again, for convenience we will often ignore these details.

We will now describe set constructors which build (new) sets from other sets. In each case, the resulting sets is well defined as long as the original sets are.

Definition 1.9 (COMBINING SETS) Let A and B be any sets.

1) The following operations construct (new) sets:

$$\begin{array}{lrcl} \textit{Set Union}: & A \cup B & \triangleq & \{x \mid x \in A \vee x \in B\} \\ \textit{Set Intersection}: & A \cap B & \triangleq & \{x \in A \mid x \in B\} \\ \textit{Set Difference}: & A \setminus B^{3} & \triangleq & \{x \in A \mid x \notin B\} \\ \textit{Symmetric Set Difference}: & A \bigtriangleup B & \triangleq & (A \setminus B) \cup (B \setminus A) \end{array}$$

2) We will also use the following notion:

$$\textit{Disjoint Sets}: \quad A, B \textit{ are disjoint} \;\triangleq\; A \cap B = \emptyset$$

We call $A \cup B$ a *disjoint union* when A and B are disjoint.

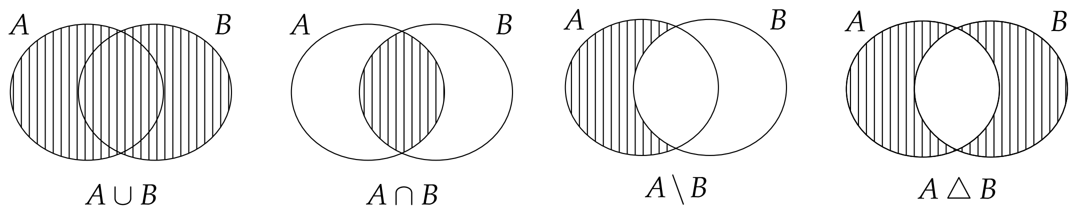
Notice that $A \cup B$ is defined to be a set, and that it is acceptable to write $\{x \mid x \in A \wedge x \in B\}$ for $A \cap B$ and $\{x \mid x \in A \wedge x \notin B\}$ for $A \setminus B$; you can use whichever is more convenient.

Moreover, we can define $A \cap B \triangleq \{x \in A \mid x \in B\}$ or $\{x \in B \mid x \in A\}$ and $A \setminus B \triangleq \{x \in A \mid x \notin B\}$, but cannot do that for $\cup$: it is *assumed* to create a set (see also Section 8).

Example 1.10 Let $A = \{1,3,5,7,9\}$ and $B = \{3,5,6,10,11\}$. Then

$$\begin{array}{rcl} A \cup B & = & \{1,3,5,6,7,9,10,11\} \\ A \cap B & = & \{3,5\} \\ A \setminus B & = & \{1,7,9\} \\ A \bigtriangleup B & = & \{1,6,7,9,10,11\} \end{array}$$

It is often helpful to illustrate these combinations of sets using *Venn diagrams*.



We will now investigate certain equalities between sets constructed from our operations.

Proposition 1.11 (PROPERTIES OF OPERATORS) *Let* A, B *and* C *be arbitrary sets. The following properties hold:*[4]

$$\begin{array}{ccc} \textit{Idempotence} & \textit{Commutativity} & \textit{Associativity} \\ A \cup A = A & A \cup B = B \cup A & A \cup (B \cup C) = (A \cup B) \cup C \\ A \cap A = A & A \cap B = B \cap A & A \cap (B \cap C) = (A \cap B) \cap C \\ & A \bigtriangleup B = B \bigtriangleup A & \end{array}$$

[3] In the literature, also the notation $A - B$ is used for set difference.

[4] We will use the notions defined here, like commutativity and associativity, also for other operations.

Empty set	*Distributivity*	*Absorption*
$A \cup \phi = A$	$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$	$A \cup (A \cap B) = A$
$A \cap \phi = \phi$	$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$	$A \cap (A \cup B) = A$
$A \triangle A = \phi$		

Proof: We will just look at the first distributivity equality;[5] some of the other cases will be set as exercises. Let A, B, and C be arbitrary sets.

By definition of equality between sets, the proof is given by the following two results:

1) $A \cup (B \cap C) \subseteq (A \cup B) \cap (A \cup C)$.

2) $(A \cup B) \cap (A \cup C) \subseteq A \cup (B \cap C)$.

To prove part (*1*), assume $x \in A \cup (B \cap C)$ for an arbitrary element x. By the definition of set union, this means that $x \in A$ or $x \in B \cap C$. By the definition of set intersection, this means that either $x \in A$, or x is in both B and C. By the distributivity of the logical 'or' over 'and', it follows that $x \in A$ or $x \in B$, and also that $x \in A$ or $x \in C$. This means that $x \in A \cup B$ and $x \in A \cup C$, and hence $x \in (A \cup B) \cap (A \cup C)$. We have shown that assuming $x \in A \cup (B \cap C)$ leads to $x \in (A \cup B) \cap (A \cup C)$, so $A \cup (B \cap C) \subseteq (A \cup B) \cap (A \cup C)$.

Alternatively, we can write:

$$\begin{aligned} x \in A \cup (B \cap C) &\Rightarrow x \in A \vee x \in B \cap C \\ &\Rightarrow x \in A \vee (x \in B \wedge x \in C) \\ &\Rightarrow (x \in A \vee x \in B) \wedge (x \in A \vee x \in C) \\ &\Rightarrow x \in A \cup B \wedge x \in A \cup C \\ &\Rightarrow x \in (A \cup B) \cap (A \cup C) \end{aligned}$$

To prove part (*2*), we must prove that $x \in (A \cup B) \cap (A \cup C)$ implies $x \in A \cup (B \cap C)$. In this case the proof is easily obtained, since it just follows the above proof in reverse (so $\Rightarrow$ gets replaced by $\Leftarrow$ and in fact becomes $\Leftrightarrow$). ■

Notice that this proof depends on the distributivity of the logical connective $\vee$ over $\wedge$.

The second proof can also be written as :

$$\begin{array}{lll} A \cup (B \cap C) & \triangleq & (\textit{by definition of} \cup) \\ \{x \mid x \in A \vee x \in (B \cap C)\} & \triangleq & (\textit{by definition of} \cap) \\ \{x \mid x \in A \vee (x \in B \wedge x \in C)\} & = & (\textit{distributivity of} \vee \textit{over} \wedge) \\ \{x \mid (x \in A \vee x \in B) \wedge (x \in A \vee x \in C)\} & \triangleq & (\textit{by definition of} \cup) \\ \{x \mid (x \in A \cup B) \wedge (x \in A \cup C)\} & \triangleq & (\textit{by definition of} \cap) \\ (A \cup B) \cap (A \cup C) & & \end{array}$$

Notice that we have used the property '$\{x \in A \mid P(x)\} = \{x \in A \mid Q(x)\}$ when $\forall x \in A\ (P(x) \Leftrightarrow Q(x))$ is evident' here.

The above proof is an example of the generality often required to prove a property about sets: it uses arbitrary sets and arbitrary elements of such a set. In contrast, to show that a property is false, it is enough to find *one* counter example. Such counter examples should be as simple as possible, to illustrate that a statement is not true with minimum effort to the reader.

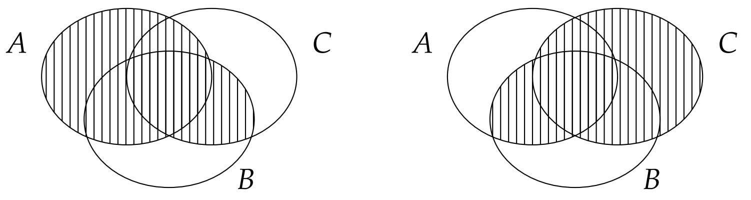
Proposition 1.12 The following statements are not always true:

1) $A \cup (B \cap C) = (A \cap B) \cup C$*;*

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

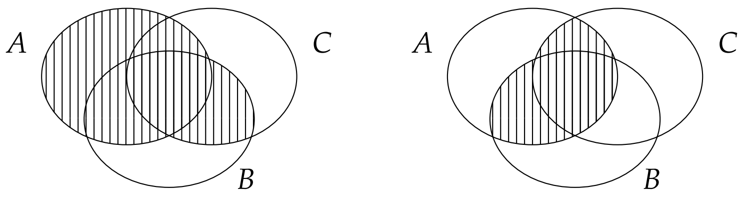
Proof: We can understand that the sets are different by looking at the Venn diagrams:

[5] Drawing a Venn diagram gives some evidence that the property is indeed true, but notice that this would not be a proof, but rather a support for understanding and intuition.



Focussing on the areas that are different and not shaded, we can construct a counter example by making sure that C has an empty intersection with A and B, and A has an empty intersection with B and C. So take $A = \{a\}$, $B = \emptyset$, $C = \{c\}$, where a and c are different objects. Then $B \cap C = \emptyset$ and $A \cup (B \cap C) = \{a\}$ whereas $A \cap B = \emptyset$ and $(A \cap B) \cup C = \{c\}$.

For part 2, we have



A counterexample to part 2 is $A = \{a\}$, $B = C = \{b\}$, where again a and b are different. Then $A \cup (B \cap C) = \{a,b\}$ and $(A \cap B) \cup (A \cap C) = \emptyset$. ■

We can show that $\cup$ and $\cap$ are *idempotent*.

Proposition 1.13 (IDEMPOTENCE) *For any set A, $A \cup A = A$ and $A \cap A = A$.*

Proof: $A \cup A = \{x \mid x \in A \vee x \in A\} = \{x \mid x \in A\} = A$;
$A \cap A = \{x \mid x \in A \wedge x \in A\} = \{x \mid x \in A\} = A$.

Notice that we have used that $\forall x\ (x \in A \vee x \in A \Leftrightarrow x \in A)$ and $\forall x\ (x \in A \wedge x \in A \Leftrightarrow x \in A)$. ■

We will now begin to explore the number of elements in a finite set. In Section 4.5 we will learn how to talk about the number of elements in an infinite set.

Definition 1.14 (CARDINALITY) Let A be a finite set. The *cardinality of* A, written $|A|$, is the number of (distinct) elements contained in A.

Remember that we defined that elements in a set are all distinct, where a and b are distinct when $a \neq b$.

Proposition 1.15 Let A and B be finite sets.
1) If A and B are disjoint, then $|A \cup B| = |A| + |B|$.
2) $|A \cup B| = |A| + |B| - |A \cap B|$.

Proof: 1) Let $|A| = n$, $|B| = m$, then there exists distinct $a_1,\ldots,a_n$, and $b_1,\ldots,b_m$ such that $A = \{a_1,\ldots,a_n\}$ and $B = \{b_1,\ldots,b_m\}$. Since A and B are disjoint, we have $a_i \neq b_j$ for all $1 \leq i \leq n$ and $1 \leq j \leq m$. Then $A \cup B = \{a_1,\ldots,a_n,b_1,\ldots,b_m\}$, which are all distinct, so $|A \cup B| = n + m = |A| + |B|$.

2) It is easy to check that

$$\begin{aligned} A &= (A \backslash B) \cup (A \cap B) \\ B &= (B \backslash A) \cup (A \cap B) \\ A \cup B &= (A \backslash B) \cup (A \cap B) \cup (B \backslash A) \end{aligned}$$

(see Exercise 10) and that unions on the right are disjoint. Then by the first part,

$$\begin{aligned} |A| &= |A \backslash B| + |A \cap B| \\ |B| &= |B \backslash A| + |A \cap B| \\ |A \cup B| &= |A \backslash B| + |A \cap B| + |B \backslash A| \end{aligned}$$

so

$$\begin{aligned} |A \cup B| &= |A \setminus B| + |A \cap B| + |B \setminus A| \\ &= |A| - |A \cap B| + |A \cap B| + |B| - |A \cap B| \\ &= |A| + |B| - |A \cap B| \end{aligned} \qquad \blacksquare$$

In Definition 1.3, we defined the notion of a subset of a set. We can create the set of all subsets of A, which is called the power set[6] of A, which is a well-defined set as well.

Definition 1.16 (POWER SET) Let A be any set. Then the collection of all subsets of A, the *power set of* A, written $\wp A$ (which are sets by definition), is a set as well and is defined as:

$$\wp A \triangleq \{X \mid X \subseteq A\}$$

Examples of power sets include:

$$\begin{aligned} \wp\{a,b\} &= \{\emptyset, \{a\}, \{b\}, \{a,b\}\} \\ \wp\emptyset &= \{\emptyset\} \\ \wp I\!N &= \{\emptyset, \\ &\quad \{0\},\{1\},\{2\},\ldots, \\ &\quad \{0,1\},\{0,2\},\ldots,\{1,1\},\{1,2\},\ldots,\{2,2\},\{2,3\},\ldots, \\ &\quad \{0,1,2\},\{0,1,3\},\ldots \\ &\quad \{2,4,6,8,\ldots\},\ldots \\ &\quad \ldots\}^{7} \end{aligned}$$

Note that the powerset of the empty set is not empty.

Let A be an arbitrary finite set. One way to list all the elements of $\wp A$ is to start with $\emptyset$, then add the sets taking one element of A at a time, then the sets taking two elements from A at a time, and so on until the whole set A is added, and $\wp A$ is complete.

Proposition 1.17 Let A be a finite set with $|A| = n$. Then $|\wp A| = 2^n$.

Proof: Consider an arbitrary set $A = \{a_1, \ldots, a_n\}$. We form a subset X of A by taking each element a_i in turn and deciding whether or not to include it in X. This gives us n independent choices between two possibilities: in X or not. We can therefore represent each subset of A by a binary number

$$b = b_1 \cdots b_n,$$

where $b_i = 1$ if $a_i \in X$, and $b_i = 0$ if $a_i \notin X$.[8] The number of different subsets we can form is therefore 2^n, the number of binary numbers representable with n bits. $\blacksquare$

Note that, in general, there are m^n ways of making n independent choices between m options.

1.4 Products

The set construct we will now consider is that of the product of two (or an arbitrary number of) sets, which forms an essential part of the definition of *relation* as discussed in the next section. If we want to describe the relationship John loves Mary, then we require a way of talking about John and Mary at the same time. We do this using the notion of an *ordered pair* $\langle a,b \rangle$[9] which is a pair of objects a and b. In our example, we have the pair $\langle \text{John}, \text{Mary} \rangle$ in the loves relation, but not necessarily the pair $\langle \text{Mary}, \text{John} \rangle$ since the love might be unrequited; so $\langle \text{Mary}, \text{John} \rangle$ is not the same as $\langle \text{John}, \text{Mary} \rangle$, hence, the order of the pair is important.

The product[10] constructor allows us to collect the ordered pairs together in one set.

[6] The name is due to the cardinality result in Proposition 1.17.

[7] We will see in Example 6.6 that we cannot effectively list this set.

[8] See also Definition 4.6.

[9] Another notation for pairs is (a,b); we use the 'pointy bracket' notation for reasons of readability.

[10] The terminology is due to the result in Proposition 1.20. In fact, we could have defined $\langle A,B \rangle$ for the set of all pairs constructed out of the sets A and B, but the product notation is universal.

Definition 1.18 (CARTESIAN PRODUCT) Let A and B be arbitrary sets.

1) An *ordered pair* of elements of sets A and B is written as $\langle a,b \rangle$.

2) The *Cartesian* (or *binary*) *product* of A and B, written $A \times B$, is a set as well and is defined as

$$A \times B \;\triangleq\; \{\langle a,b \rangle \mid a \in A \wedge b \in B\}$$

We will write A^2 for $A \times A$.

3) Equality on elements of $A \times B$ is defined as: $\forall a,b,c,d\,(\langle a,b \rangle =_{A\times B} \langle c,d \rangle \triangleq a =_A c \wedge b =_B d)$.

Notice that $\times$ is not commutative, since, in general, $A \times B \neq B \times A$, and that $\langle a,b \rangle \neq \{a,b\}$.

Example 1.19 Simple examples of Cartesian products include:

- The coordinate system of real numbers $I\!R^2$: points are described by their coordinates $\langle x,y \rangle$;
- (Heterosexual) computer marriage bureau: let M be the set of men registered and W the set of women, then the set of all possible matches is $M \times W$;

The use of the symbol $\times$ to denote the set of all pairs (and of the terminology *product*) is motivated by the next result.

Proposition 1.20 Let A and B be finite sets. Then $|A \times B| = |A| \times |B|$.

Proof: Let A and B be arbitrary sets with $A = \{a_1,\dots,a_m\}$ and $B = \{b_1,\dots,b_n\}$. For each element a_i in A there are m choices of element b_j in B to form a pair $\langle a_i,b_j \rangle$. The total number of possible pairs is therefore $n \times m$. ■

We can extend the notion of Cartesian product to the higher case.

Definition 1.21 (n-ARY PRODUCT) *1*) For any $n \geq 1$, an n-tuple is a sequence $\langle a_1,\dots,a_n \rangle$ of n objects.

2) Let $A_1,\dots,A_n$ be arbitrary sets. The n-ary product of the A_i, written $A_1 \times \cdots \times A_n$ or $\prod_{i=1}^{n} A_i$, is $\{\langle a_1,\dots,a_n \rangle \mid \forall 1 \leq i \leq n\; (a_i \in A_i)\}$.

3) The n-ary product of As is written A^n, with A^2 the Cartesian product (Definition 1.18).

In the first part, again, the order of the a_i matters.

Example 1.22 • The three-dimensional space of real numbers $I\!R^3$ is an example of a ternary product.

- Let A_1, …, A_n be a collection of finite sets. Then $|A_1 \times \cdots \times A_n| = |A_1| \times \cdots \times |A_n|$.
- Notice that we can now form the product of three sets in three different ways:

$$A \times B \times C \qquad (A \times B) \times C \qquad A \times (B \times C)$$

These sets are all different, so the set-operator $\times$ is not associative, whereas multiplication $\times$ on numbers is. There exists however a clear correspondence between the elements $\langle a,b,c \rangle$ and $\langle \langle a,b \rangle, c \rangle$ and $\langle a, \langle b,c \rangle \rangle$, and so these sets are in some sense equivalent; see also Exercise 39.

1.5 Partitions

If we divide a set into non-overlapping chunks we get a partition of the set.

Definition 1.23 For any set S, a *partition of* S is a family $A_1,\dots,A_n$ of subsets of S such that: (1) each A_i is non-empty; (2) the A_i cover S; and (3) the A_i are pairwise disjoint.

$$\begin{array}{rcl} \forall 1 \leq i \leq n\,(A_i & \neq & \emptyset) \\ S & = & A_1 \cup \cdots \cup A_n \\ \forall 1 \leq i,j \leq n\,(i \neq j \Rightarrow & A_i \cap A_j = \emptyset) & \end{array}$$

It is sometimes convenient to write $A_1 \cup \cdots \cup A_n$ as $\cup_{i=1}^{n} A_i$.

For example, cars can be partitioned into *makes* and integers can be partitioned into *even* and *odd*. We shall return to partitions when discussing equivalence relations.

Set partitioning is strongly related to the pigeonhole principle, which can be stated as follows. Suppose that m objects are to be placed in n pigeonholes, where $m > n$; then some pigeonhole will have to contain more than one object.

The pigeonhole principle is an example of a property that needs a proof by contradiction.

Theorem 1.24 (THE PIGEONHOLE PRINCIPLE) *If a set of n distinct objects is partitioned into k subsets, where $0 < k < n$, then at least one subset must contain at least two elements.*

Proof: Let $|A| = n$, and $A_1,\ldots,A_k$ be a partition of A; in particular, each A_i is not empty. Assume that each A_i has only one element. Since $A = \cup_{i=1}^k A_i$ is a partition of A, by Proposition 1.15 we know that $|A| = |\cup_{i=1}^k A_i|$, with the A_i pairwise disjoint, so $|\cup_{i=1}^k A_i| = \Sigma_{i=1}^k |A_i|$. Notice that then $n = |A| = |\cup_{i=1}^k A_i| = \Sigma_{i=1}^k |A_i| = k$. It is impossible for $k < n$ and $n = k$, so we have a contradiction. So there is at least one A_i that has at least two elements. ■

The property also holds when we drop the condition that $A_1,\ldots,A_k$ is a partition, but just demand that $A = \cup_{i=1}^k A_i$ with the A_i pairwise disjoint, and allow each A_i to be empty.

Exercises

Exercise 1 *Show that $A = B$ if and only if $\forall x\ (x \in A \Leftrightarrow x \in B)$.*

Exercise 2 *Draw Venn diagrams representing 2 and 3 possibly overlapping sets. How many regions are there in each case? Can you draw an appropriate diagram for 4 sets? Again, how many regions does it contain?*

Exercise 3 *Let $A = \{1,2,3,4\}$, $B = \{3,4,6\}$, $C = \{1,6\}$. Calculate the following: (1) $B \cup C$ (2) $A \cap (B \cup C)$ (3) $(A \cap B) \cup (A \cap C)$.*

Exercise 4 *Let $A = \{1,3,5\}$ and $B = \{2,3\}$. Write down explicit sets for (1) $A \cup B$ and $A \cap B$; (2) $A \setminus B$ and $B \setminus A$; (3) $(A \cup B) \setminus B$ and $(A \setminus B) \cup B$; (4) $A \triangle B$ and $B \triangle A$; (5) $A \times B$, $A \times \emptyset$ and $B \times A$; (6) $\wp A$.*

Exercise 5 *Let $A = \{\{a\},\{b\}\}$ and $B = \{a,b,\{a\}\}$. Determine $A \cap B$, $A \cup B$, $\wp B$, $A \cap \wp B$, $A \times B$, $(A \times B) \cap (B \times A)$ and $A \triangle B$.*

Exercise 6 *Determine whether the following statements are true or false. (1) $\{x\} \subseteq \{x\}$; (2) $\{x\} \in \{x\}$; (3) $\{x\} \in \{x,\{x\}\}$; (4) $\{x\} \subseteq \{x,\{x\}\}$; (5) $\emptyset \in \emptyset$; (6) $\emptyset \subseteq \emptyset$; (7) $\emptyset \subseteq \{\emptyset\}$.*

Exercise 7 *Of 100 students, 35 play football, 36 play tennis and 24 play cricket. 13 play football and tennis, 2 play football and cricket but never tennis, 12 play tennis and cricket, while 4 play in all three games. How many students do not play these games? Justify your answer.*

Exercise 8 *Let A, B and C be sets. Decide whether each of the following statements is true or false. Justify your answer either with a formal proof or a counter-example.*

1) $A \cup (B \cap C) = (A \cap B) \cup C$
2) $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$
3) $A \cup (B \cap C) = (A \cap B) \cup (A \cap C)$
4) $A \setminus (B \cup C) = (A \setminus B) \cap (A \setminus C)$
5) $A \setminus (B \cup C) = (A \setminus B) \cup (A \setminus C)$
6) $A \cap (B \setminus C) = (A \cap B) \setminus (A \cap C)$
7) $A \triangle (B \cap C) = (A \triangle B) \cap (A \triangle C)$
8) $A \cap (B \triangle C) = (A \cap B) \triangle (A \cap C)$

Exercise 9 *Show that if A and B are disjoint finite sets, then $|A \cup B| = |A| + |B|$.*

Exercise 10 *Show that*
$$\begin{aligned} A &= (A \setminus B) \cup (A \cap B) \\ A \cup B &= (A \setminus B) \cup (A \cap B) \cup (B \setminus A) \end{aligned}$$
and that these unions are disjoint.

* *Exercise 11* *Let A, B be finite sets. Recall that $|A \cup B| = |A| + |B| - |A \cap B|$. Find a similar formula for $|A \cup B \cup C|$, and justify your answer.*

Exercise 12 *Calculate $\wp\{0\}$, $\wp\{0,\{0\}\}$, $\wp\{0,\{0\},\{0,\{0\}\}\}$.*

* *Exercise 13* *Let A be a finite set, and $A_1,\ldots,A_k$ be a partition of A. Show that $|A| = \Sigma_{i=1}^k |A_i|$.*

2 Relations

We wish to capture the concept of objects being related: for example, John *loves* Mary; $2 < 5$; two programs P and Q are equal. Such properties can be expressed using relations.

Definition 2.1 (RELATION) *1)* A *relation* R between two sets A and B, or *from* A *to* B, is a subset of the binary product of A and B, so $R \subseteq A \times B$. We use $R, S, \ldots$ to range over relations.

2) If $R \subseteq A \times B$, we say that R *has type* $A \times B$.

3) If $R \subseteq A^2$, we call R a *binary relation on* A.

Instead of $\langle a,b \rangle \in R$, we often write $a\,R\,b$.

Notice that, since a relation is a subset of a product set, equality between relations is actually equality between sets.

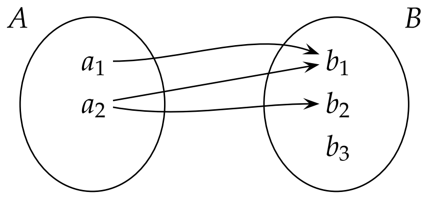
In general, there will be many relations on any set. A relation does not have to be meaningful; any subset of a Cartesian product forms a relation. For example, for $A = \{a,b\}$, there are *sixteen* binary relations on A:

$$\begin{array}{llll}
\emptyset & \{\langle b,b\rangle\} & \{\langle a,a\rangle,\langle a,b\rangle\} & \{\langle a,a\rangle,\langle a,b\rangle,\langle b,b\rangle\} \\
\{\langle a,a\rangle\} & \{\langle a,b\rangle,\langle b,a\rangle\} & \{\langle a,a\rangle,\langle b,a\rangle\} & \{\langle a,a\rangle,\langle b,a\rangle,\langle b,b\rangle\} \\
\{\langle a,b\rangle\} & \{\langle a,b\rangle,\langle b,b\rangle\} & \{\langle a,a\rangle,\langle b,b\rangle\} & \{\langle a,b\rangle,\langle b,a\rangle,\langle b,b\rangle\} \\
\{\langle b,a\rangle\} & \{\langle b,a\rangle,\langle b,b\rangle\} & \{\langle a,a\rangle,\langle a,b\rangle,\langle b,a\rangle\} & \{\langle a,a\rangle,\langle a,b\rangle,\langle b,a\rangle,\langle b,b\rangle\}
\end{array}$$

Note that there are four elements (possible pairs) in $A^2 = \{\langle a,a\rangle,\langle a,b\rangle,\langle b,a\rangle,\langle b,b\rangle\}$; each relation is a subset of A^2, so there are $|\wp A^2| = 16$ possibilities. Notice that there are only four subsets of $\{a,b\}$: $\emptyset$, $\{a\}$, $\{b\}$, and $\{a,b\}$.

However, listing ordered pairs can get tedious and hard to follow. For binary relations $R \subseteq A \times B$ we have several other representations.

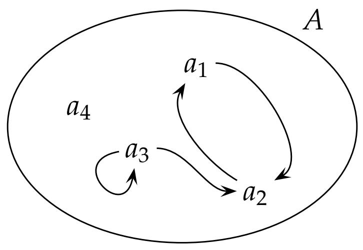
(*Diagram*): Let $A = \{a_1,a_2\}$, $B = \{b_1,b_2,b_3\}$, and $R = \{\langle a_1,b_1\rangle,\langle a_2,b_1\rangle,\langle a_2,b_2\rangle\}$. We can represent R by the diagram:



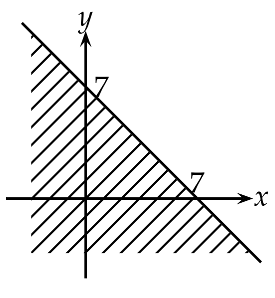
(*Directed Graph*): In case R is a binary relation on A, we can also use a *directed* graph, which consists of a set of nodes corresponding to the elements in A, joined by arrowed lines indicating the relationship between the elements. For example, let

$$\begin{array}{rcl} A & = & \{a_1,a_2,a_3,a_4\} \\ R & = & \{\langle a_1,a_2\rangle,\langle a_2,a_1\rangle,\langle a_3,a_2\rangle,\langle a_3,a_3\rangle\} \end{array}$$

The directed graph of this relation is given to the right. Notice that the direction of the arrows matters. (It is, of course, still possible to use a diagram where the source and target sets are drawn separately as above.)



(*Special representations*): Some special ways of drawing certain important relations exist. For example, we can represent a binary relation on $I\!R$ as an area in the plane. The diagram to the right represents the relation R defined by $x\,R\,y \triangleq x + y \leq 7$.



(*Matrix*): Take $A = \{a_1,\ldots,a_m\}$ and $B = \{b_1,\ldots,b_n\}$. We can represent R by an $m \times n$ matrix M

of booleans True and False. For $i = 1,\ldots,m$ and $j = 1,\ldots,n$, define

$$M(i,j) \;=\; \begin{cases} \textsf{True}, & (\textit{if } a_i \, R \, b_j) \\ \textsf{False}, & (\textit{otherwise}) \end{cases}$$

where $M(i,j)$ is the usual notation for position on the ith row and jth column of the matrix. For example, if $A = \{a_1,a_2\}$, $B = \{b_1,b_2,b_3\}$ and $R = \{\langle a_1,b_1\rangle,\langle a_2,b_1\rangle,\langle a_2,b_2\rangle\}$ as before, then the matrix is

$$\begin{pmatrix} \textsf{True} & \textsf{False} & \textsf{False} \\ \textsf{True} & \textsf{True} & \textsf{False} \end{pmatrix}$$

It is also common to use the elements 0, 1, or F, T, instead of False, True.

Just as for products, we can extend the definition of a binary relation to an n-ary relation, for any n.

Definition 2.2 A *n-ary relation* between the sets A_1, ..., A_n is a subset of the n-ary product $\prod_{i=1}^{n} A_j$. The definition of a 2-ary relation is the same as that of a binary relation given in Definition 2.1. A predicate over the set A is a 1-ary relation: that is, a subset of A.

Example 2.3 1) The set $\{p \in \mathit{IN} \mid p \textit{ is prime}\}$ is a unary relation on IN.

2) The set $\{\langle x,y,z\rangle \in \mathit{IR}^3 \mid x^2+y^2+z^2 = 1\}$ is a ternary relation on IR, which describes the surface of the unary sphere with centre $\langle 0,0,0\rangle$.

2.1 Constructing relations

Just as for sets, we may construct relations from others. We just give the definitions for binary relations; it is easy to extend the definitions to higher numbers.

Definition 2.4 (Basic Relation Operators) Let R, $S \subseteq A \times B$. We define the relations $R \cup S$, $R \cap S$ and $\overline{R}$, all with type $A \times B$, and R^{-1} with type $B \times A$, by:

$$\begin{array}{lrcl}
\textit{Relation Union}: & R \cup S & \triangleq & \{\langle a,b\rangle \in A \times B \mid \langle a,b\rangle \in R \vee \langle a,b\rangle \in S\} \\
\textit{Relation Intersection}: & R \cap S & \triangleq & \{\langle a,b\rangle \in A \times B \mid \langle a,b\rangle \in R \wedge \langle a,b\rangle \in S\} \\
\textit{Relation Complement}: & \overline{R} & \triangleq & \{\langle a,b\rangle \in A \times B \mid \langle a,b\rangle \notin R\} \\
\textit{Inverse relation}: & R^{-1} & \triangleq & \{\langle b,a\rangle \in B \times A \mid a \, R \, b\}
\end{array}$$

Example 2.5 Let R and S be binary relations on $\{1,2,3,4\}$ defined by:

$$\begin{aligned} R &= \{\langle 1,2\rangle,\langle 2,3\rangle,\langle 3,4\rangle,\langle 4,1\rangle,\langle 4,4\rangle\} \\ S &= \{\langle 1,2\rangle,\langle 2,1\rangle,\langle 3,4\rangle,\langle 4,3\rangle\} \end{aligned}$$

We can construct the following relations:

$$\begin{aligned}
R \cup S &= \{\langle 1,2\rangle,\langle 2,3\rangle,\langle 3,4\rangle,\langle 4,1\rangle,\langle 4,4\rangle,\langle 2,1\rangle,\langle 4,3\rangle\} \\
R \cap S &= \{\langle 1,2\rangle,\langle 3,4\rangle\} \\
\overline{R} &= \{\langle 1,1\rangle,\langle 1,3\rangle,\langle 1,4\rangle,\langle 2,1\rangle,\langle 2,2\rangle,\langle 2,4\rangle,\langle 3,1\rangle,\langle 3,2\rangle,\langle 3,3\rangle,\langle 4,2\rangle,\langle 4,3\rangle\} \\
R^{-1} &= \{\langle 2,1\rangle,\langle 3,2\rangle,\langle 4,3\rangle,\langle 1,4\rangle,\langle 4,4\rangle\}
\end{aligned}$$

Notice that we have overloaded notation: $R \cup S$ and $R \cap S$ denote relation union and intersection respectively when R and S are viewed as relations, and set union and intersection when viewed as sets. To form a relation union or intersection, the relations must be of the same type. In contrast, we can form the union and intersection of arbitrary sets. It should be clear from the context which interpretation we intend.

It is easy to verify that if we take the inverse of the inverse of a relation, we recover the original: $(R^{-1})^{-1} = R$. Note that the inverse is only defined for binary relations and that any binary relation has an inverse. Sometimes the inverse is indicated by reversing an infix relation: '$>$' is the inverse of '$<$', etc.

The inverse of a relation should not be confused with its complement: for example, the inverse of 'is a parent of' is 'is a child of';

$$x \text{ parent of } y \;\Leftrightarrow\; y \text{ child of } x$$

whereas the complement of 'is a parent of' is 'is not a parent of';

$$x \;\overline{\text{parent of}}\; y \;\Leftrightarrow\; \neg(x \text{ parent of } y)$$

As well as inheriting structure from sets, relations have operations of their own which do not apply to sets in general, like identity and composition, which we will now define.

Definition 2.6 (IDENTITY RELATION) Given a set A, the *identity relation* on A, written id_A, is the binary relation on A defined by

$$id_A \;\triangleq\; \{\langle x,y\rangle \in A^2 \mid x =_A y\}$$

Definition 2.7 (COMPOSITION OF RELATIONS) Given $R \subseteq A \times B$ and $S \subseteq B \times C$, then the *composition of R with S*, written $R \circ S$, is defined by

$$R \circ S \;\triangleq\; \{\langle a,c\rangle \in A \times C \mid \exists b \in B\ (a\,R\,b \wedge b\,S\,c)\}$$

We will often write $a\,R\,b\,S\,c$ for $a\,R\,b \wedge b\,S\,c$.

The notation $R \circ S$ can be read as R *composed with* S. Notice that the relation $R \circ S$ is only defined if the types of R and S match up.

Example 2.8
- We can define the set 'grandparent of $\triangleq$ parent of $\circ$ parent of' by:

$$x \text{ grandparent of } y \;\triangleq\; \exists z\ (x \text{ parent of } z \wedge z \text{ parent of } y)$$

- Using the relations R and S from Example 2.5, we have

$$\begin{aligned} R \circ S &= \{\langle 1,1\rangle, \langle 2,4\rangle, \langle 3,3\rangle, \langle 4,2\rangle, \langle 4,3\rangle\} \\ S \circ R &= \{\langle 1,3\rangle, \langle 2,2\rangle, \langle 3,1\rangle, \langle 3,4\rangle, \langle 4,4\rangle\} \end{aligned}$$

2.2 Equalities between Relations

Recall from Proposition 1.11 that we can prove certain equalities between sets constructed through set operations. We can also show similar equalities between relations constructed through operations on relations.

Proposition 2.9 1) If $R \subseteq A \times B$, then $id_A \circ R = R = R \circ id_B$.

2) Composition is associative: for arbitrary relations $R \subseteq A \times B$ and $S \subseteq B \times C$ and $T \subseteq C \times D$, we have $R \circ (S \circ T) = (R \circ S) \circ T$.

Proof: The proof of part (*1*) is simple and left as Exercise 19; we prove part (*2*). Let R, S, and T be relations specified in the proposition, and let $\langle x,u\rangle$ be an arbitrary member of $(R \circ S) \circ T$. We show that $\langle x,u\rangle \in R \circ (S \circ T)$ as follows:

$$\begin{aligned} x\,(R \circ S) \circ T\,u &\Rightarrow \exists z\ (x\,(R \circ S)\,z \wedge z\,T\,u) \\ &\Rightarrow \exists z\ (\exists y\ (x\,R\,y \wedge y\,S\,z) \wedge z\,T\,u) \\ &\Rightarrow \exists z,y\ (x\,R\,y \wedge y\,S\,z \wedge z\,T\,u) \\ &\Rightarrow \exists y\ (x\,R\,y \wedge \exists z\ (y\,S\,z \wedge z\,T\,u)) \\ &\Rightarrow \exists y\ (x\,R\,y \wedge y\,(S \circ T)\,u) \\ &\Rightarrow x\,R \circ (S \circ T)\,u \end{aligned}$$

We have shown that $(R \circ S) \circ T \subseteq R \circ (S \circ T)$. The reverse, showing that $R \circ (S \circ T) \subseteq (R \circ S) \circ T$ can be proved in a similar way. ■

We do not really 'need' operators like '$\circ$', since we can always replace them by their definitions and work directly with their components (using predicate logic, etc.), but the operators are a convenient shorthand and laws such as associativity give us something of the ease and calculating power of ordinary arithmetic.

Proposition 2.10 Let R and S be arbitrary binary relations on A. In general,

1) $R \neq R^{-1}$;

2) $R \circ S \neq S \circ R$ (so composition is not commutative);

3) $R \circ R^{-1} \neq id_A$.

Proof: Just as for Proposition 1.12, the way to prove that a property does not hold is to provide a counter example. For a counter example to part (*1*), take $A = \{b,a\}$ with $a \neq b$, and the relation $R = \{\langle a,b\rangle\}$. Then $R^{-1} = \{\langle b,a\rangle\}$ which is plainly different from R.

To show that composition is not commutative (part (*2*)), we must find R and S such that $R \circ S \neq S \circ R$. Well, take $A = B = \{a,b\}$ with $a \neq b$, $R = \{\langle a,a\rangle\}$, and $S = \{\langle a,b\rangle\}$. Then $R \circ S = \{\langle a,b\rangle\}$, but $S \circ R = \emptyset$.

Part (*3*) is left as Exercise 19. ■

* *Example 2.11* Members of staff occupy various rooms (possibly more than one). They have various keys, and these keys open various rooms. We model this by defining three sets – staff, key, room – and three binary relations:

$$\begin{aligned} \mathsf{occupies} &\subseteq \mathsf{staff} \times \mathsf{room} \\ \mathsf{hasKey} &\subseteq \mathsf{staff} \times \mathsf{key} \\ \mathsf{opens} &\subseteq \mathsf{key} \times \mathsf{room} \end{aligned}$$

We would like to know which staff members can open which rooms. This will be a relation

$$\mathsf{canOpen} \subseteq \mathsf{staff} \times \mathsf{room}$$

Plainly s canOpen r if and only if s hasKey k and k opens r, for some key k. But this means that

$$\mathsf{canOpen} = \mathsf{hasKey} \circ \mathsf{opens}.$$

Notice that a member of staff may be able to open a room using more than one key.

A question of interest is whether staff members can open the rooms they occupy. In other words, is it true that

$$\forall s,r\ (s\ \mathsf{occupies}\ r \Rightarrow s\ \mathsf{canOpen}\ r)$$

This is equivalent to asking whether occupies $\subseteq$ canOpen.

Equivalence Relations

We give some universal definitions of the properties a relation might satisfy.

Definition 2.12 Let R be a binary relation on A. Then

$$\begin{aligned} R \textit{ is reflexive} &\triangleq \forall x \in A\ (\langle x,x\rangle \in R) \\ R \textit{ is symmetric} &\triangleq \forall \langle x,y\rangle \in A^2\ (\langle x,y\rangle \in R \Rightarrow \langle y,x\rangle \in R) \\ R \textit{ is transitive} &\triangleq \forall \langle x,z\rangle \in A^2\ (\exists y \in A\ (\langle x,y\rangle \in R \wedge \langle y,z\rangle \in R) \Rightarrow \langle x,z\rangle \in R) \end{aligned}$$

Relations with these properties occur naturally: the equality relation on sets is reflexive, symmetric and transitive; the relations $\leq$ on numbers and $\subseteq$ on sets are reflexive and transitive, but not symmetric; and the relation $<$ on numbers is transitive, but not reflexive nor symmetric.

Alternatively, we can write:

$$\begin{aligned} R \textit{ is reflexive} &\triangleq \forall x \in A\ (x\,R\,x) \\ R \textit{ is symmetric} &\triangleq \forall x,y \in A\ (x\,R\,y \Rightarrow y\,R\,x) \\ R \textit{ is transitive} &\triangleq \forall x,z \in A\ (\exists y \in A\ (x\,R\,y \wedge y\,R\,z) \Rightarrow x\,R\,z) \end{aligned}$$

Another way to define these relations is in terms of the operations on relations introduced in the previous section.

Proposition 2.13 Let R be a binary relation on A.

1) R is reflexive if and only if $id_A \subseteq R$.

2) R is symmetric if and only if $R = R^{-1}$.

3) R is transitive if and only if $R \circ R \subseteq R$.

Proof: The proof is easy and is left as Exercise 21. ■

Sometimes we would like to consider objects 'the same' even when they are not (syntactically) identical, like 7 and $3+4$; we then call these objects *equivalent*. We can think of an equivalence relation as a weak equality: if R is an equivalence relation, then $a\ R\ b$ means that a and b are in some sense indistinguishable, but not necessarily identical as objects or expressions.

Definition 2.14 Let A be a set and R a binary relation on A. The relation R is an *equivalence* relation when R is reflexive, symmetric, and transitive. We often write $a \sim_R b$ for $a\ R\ b$ when R is an equivalence, or just $a \sim b$ when it is clear or unimportant which R is intended.

In words, we can say that R is an equivalence relation when: (1) we cannot distinguish an element from itself; (2) if x is indistinguishable from y, then so is y from x; *and* (3) if x is indistinguishable from y and y is indistinguishable from z, then so is x from z.

We can use an equivalence relations to partition a set in a natural way into disjoint subsets such that the elements in these subsets are related and equivalent to each other.

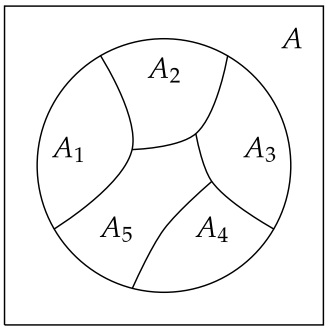
Definition 2.15 (EQUIVALENCE CLASSES) Let R be an equivalence relation on A. For any $a \in A$, the equivalence class of a with respect to R, denoted $[a]_R$, is defined as

$$[a]_R \ \triangleq\ \{x \in A \mid a \sim_R x\}.$$

We often write $[a]$ instead of $[a]_R$ when the relation R is apparent.

The set of equivalence classes is sometimes called the *quotient set* A/R (or $A/{\sim_R}$).

We can draw the separation of a set into equivalence classes in a diagram, as for example:



In this case, there are five equivalence classes, illustrated by the five disjoint subsets. In fact, the equivalence classes always separate the elements into disjoint subsets that cover the whole of the set, as the following proposition states formally.

Example 2.16 1) Given $n \in I\!N$, the binary relation R_n on $\mathbb{Z}$ defined by

$$R_n \ \triangleq\ \{\langle a,b\rangle \in \mathbb{Z}^2 \mid \exists q \in \mathbb{Z}\ (q \times n = b-a)\}$$

is an equivalence relation. The set $\mathbb{Z}/R_n$ represents the integers modulo n.

2) The binary relation S on the set $\mathbb{Z} \times I\!N \setminus \{0\}$ defined by $\langle z_1,n_1\rangle\ S\ \langle z_2,n_2\rangle$ when $z_1 \times n_2 = z_2 \times n_1$ is an equivalence relation. We can write S as:

$$S \ \triangleq\ \{\langle\langle z_1,n_1\rangle,\langle z_2,n_2\rangle\rangle \in (\mathbb{Z} \times I\!N \setminus \{0\})^2 \mid z_1 \times n_2 = z_2 \times n_1\}$$

This is the usual representation of the rational numbers (see also Definition 3.4).

3) For any set A, the equality relation $=_A$ is an equivalence relation.

4) For any set A, the identity relation id_A is an equivalence relation.

5) Given a set Student and a map age : Student $\Rightarrow I\!N$, the binary relation sameage on Student defined by

$$s_1 \ \mathsf{sameage}\ s_2 \ \triangleq\ \mathsf{age}(s_1) = \mathsf{age}(s_2)$$

is an equivalence relation.

6) Looking ahead, in Definition 4.25 we will define a relation between sets which characterises when (finite and infinite) sets have the same number of elements. Proposition 4.28 shows that this is an equivalence relation.

Proposition 2.17 Let R be an equivalence relation on A. Then the set $\{[a]_R \mid a \in A\}$ forms a partition of A.

Proof: We check the conditions of Definition 1.23:

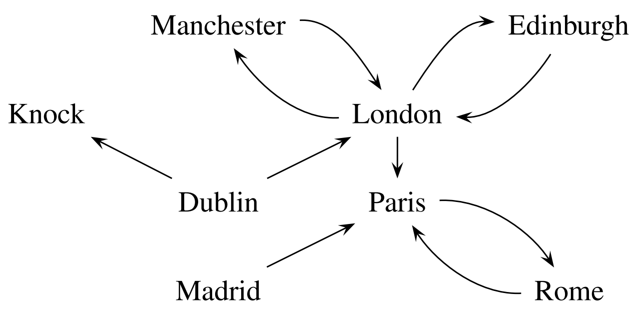
- each $[a]_R$ is non-empty;
- the classes cover A: that is, $A = \cup_{a\in A}[a]_R$;
- the classes are disjoint (or equal): $\forall a,b \in A\ ([a]_R \cap [b]_R \neq \emptyset \Rightarrow [a]_R = [b]_R)$.

Take $a \in A$, then $a \sim_R a$ by reflexivity and so $a \in [a]_R$ and $[a]_R$ is not empty. Then also $a \in \cup_{a\in A}[a]_R$, for every $a \in A$, and hence the classes cover A.

Assume $[a]_R \cap [b]_R \neq \emptyset$, and let $x \in [a]_R \cap [b]_R$. By definition, $a \sim_R x$ and $b \sim_R x$; by symmetry we have also $x \sim_R b$. To show $[b]_R \subseteq [a]_R$, take $v \in [b]_R$; then by definition $b \sim_R v$. Now $a \sim_R x$, $x \sim_R b$ and $b \sim_R v$, so $a \sim_R v$ by transitivity. Therefore $v \in [a]_R$ and so $[b]_R \subseteq [a]_R$. Using a similar argument, we can show $[a]_R \subseteq [b]_R$; so we have $[a]_R = [b]_R$. ■

2.3 Transitive Closure

Consider the following situation. There are various flights between various cities. For any two cities, we wish to know whether it is possible to fly from one to the other allowing for changes of plane. We can model this by defining a set City of cities and a binary relation R such that $a\,R\,b$ if and only if there exists a direct flight from a to b. This relation may be represented as a directed graph with the cities as nodes, as in the following example:



Now define the relation R^+ by: $a\,R^+\,b$ when there exists a trip from a to b, perhaps using many flights. Then clearly $a\,R^+\,b$ when there exists some path from a to b in the directed graph. For instance, there exists a path from Manchester to Rome, but no path from Rome to Manchester. We would like to calculate R^+ from R. Such a relation is called the *transitive closure of* R, since it is clearly transitive; it is in fact a special relation in the sense that it is the smallest transitive relation containing R.

Definition 2.18 (TRANSITIVE CLOSURE) We define R^n, with $n \geq 1$, as follows:

$$\begin{array}{lcll} R^1 & \triangleq & R & \\ R^2 & \triangleq & R \circ R & \\ R^3 & \triangleq & R \circ R^2 = R^2 \circ R, & (\circ \textit{ is associative}) \\ & \vdots & & \\ R^n & \triangleq & R \circ R^{n-1} = R \circ \cdots \circ R, & (n \textit{ times}) \\ & \vdots & & \end{array}$$

and define the *transitive closure* of R, R^+, by:

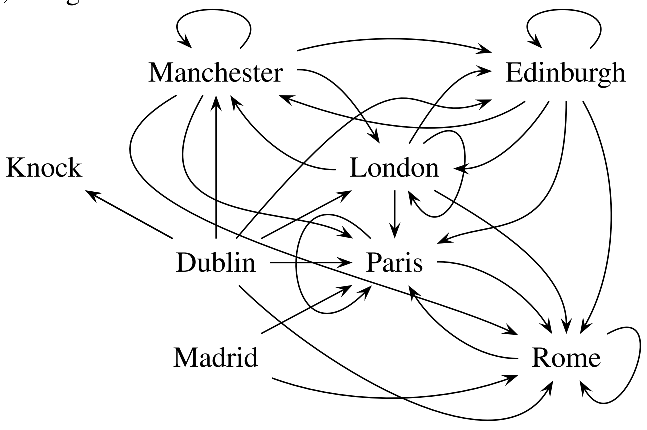
$$R^+ \ \triangleq\ R \cup R^2 \cup \cdots \cup R^n \cup \cdots \ =\ \cup_{i\geq 1} R^i$$

Therefore, we have $a\,R^+\,b$ if and only if there exists $n \geq 1$ such that $a\,R^n\,b$. The transitive closure of a binary relation R on A always exists.

We can express the relation $R^+ \subseteq A^2$ in terms of R also informally: we have $a\,R^+\,b$ when there exists some $n \geq 1$, and a 'path' of length n from a to b, i.e. there are $c_0,\ldots,c_n \in A$ such that $c_i\,R\,c_{i+1}$ for $0 \leq i < n$, and $a = c_0$ and $b = c_n$.

$$a = c_0 \ \wedge\ \exists c_o,\ldots,c_n\ (c_0\,R\,c_1\,R \cdots R\,c_{n-1}\,R\,c_n)\ \wedge\ c_n = b$$

For the above example, this gives:



To cast more light on R^+, we can understand it also through the following construction. Let R be finite, and imagine that we want to make R transitive in the most economical fashion. If R is already transitive, we need do nothing. Otherwise, there exists a, b, $c \in A$ such that $a\ R\ b$ and $b\ R\ c$, but not $a\ R\ c$; we then add the pair $\langle a,c\rangle$ to the relation, which is now in some sense closer to being transitive. We carry on doing this until there are no more pairs to add and obtain a transitive relation. Anything we have added to R was forced upon us by the requirement of transitivity alone, so we have obtained the smallest possible transitive relation containing R.

In case A is finite, we need not consider paths of length greater than $n = |A|$ since they will involve repeats (visiting the same node of the graph twice; this follows from the Pigeonhole Principle). So R^{n+1} is already included in $R \cup R^2 \cup \cdots \cup R^n$ and we need not calculate further as we have found R^+. In fact, we often do not have to go as far as $|A|$. In the airline example at the beginning of the section, there are 8 cities, but the longest paths without repeats are of length 3. Thus we compute R, R^2, R^3 and find that $R^4 \subseteq R \cup R^2 \cup R^3$, so that we can stop.

It is sometimes useful to 'reverse' the process of finding the transitive closure. In other words, given a transitive relation R, the task is to find a smallest S such that $S^+ = R$. The benefit is that S is smaller, while in some sense having the same information content as R, since R can be reconstructed from S; in general, there can be many solutions for S. We will return to this problem in the easier setting of partial orders in Section 7.

Exercises

Exercise 14 Let R and S be binary relations on $\{1,2,3,4\}$ *such that*

$$\begin{aligned} R &= \{\langle 1,2\rangle,\langle 2,3\rangle,\langle 3,4\rangle,\langle 4,1\rangle,\langle 4,4\rangle\} \\ S &= \{\langle 1,2\rangle,\langle 2,1\rangle,\langle 3,4\rangle,\langle 4,3\rangle\} \end{aligned}$$

Give $R \cup S$, $R \cap S$, *and* $\overline{R}$.

Exercise 15 Let $A = \{1,2,3,4\}$, $B = \{a,b,c,d\}$, $C = \{\alpha,\beta,\gamma\}$, *and let*

$$\begin{aligned} R &= \{\langle 1,a\rangle,\langle 2,d\rangle,\langle 3,a\rangle,\langle 3,b\rangle,\langle 3,d\rangle\} \\ S &= \{\langle b,\alpha\rangle,\langle b,\gamma\rangle,\langle c,\beta\rangle,\langle d,\gamma\rangle\} \end{aligned}$$

Give the diagram and matrix representations of R and S. For the following relations, either list their elements and give their types, or explain why they are not well defined:

$$R^{-1} \qquad \overline{S} \qquad R \cup S \qquad R \circ S$$

Exercise 16 If $|A| = n$, *how many binary relations are there on* A*? How many of these are reflexive?*

Exercise 17 If $|A| = k$ *and* $|B| = m$, *how many relations are there between* A *and* B*? If in addition* $|C| = n$, *how many ternary relations are there in* $A \times B \times C$*?*

Exercise 18 Let R, $S \subseteq A^2$. *Prove that the following statements are true:*

1) If $R \subseteq S$ *then* $R^{-1} \subseteq S^{-1}$;

2) $(R \cap S)^{-1} = R^{-1} \cap S^{-1}$;

3) $(R \cup S)^{-1} = R^{-1} \cup S^{-1}$;

4) $(\overline{R})^{-1} = \overline{R^{-1}}$;

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

Exercise 19 *1) Show that if* $R \subseteq A \times B$, *then* $id_A \circ R = R = R \circ id_B$.

2) Show that, in general, $R \circ R^{-1} \neq id_A$.

Exercise 20 *Let* $A = \{1,2,3,4\}$ *and let R be a binary relation on A.*

1) If R is reflexive, what ordered pairs must belong to R?

2) If R is symmetric, what ordered pairs must belong to R?

3) If $\langle 1,2\rangle, \langle 3,1\rangle \in R$ *and R is symmetric, then what other ordered pairs must belong to R?*

4) Is the relation $\{\langle 1,4\rangle\}$ *transitive?*

Exercise 21 *Let R be a binary relation on A. Show that*

1) R is reflexive if and only if $id_A \subseteq R$.

2) R is symmetric if and only if $R = R^{-1}$.

3) R is transitive if and only if $R \circ R \subseteq R$.

Exercise 22 *1) Is symmetric difference* $\triangle$ *idempotent? Explain your answer.*

2) Show by means of Venn diagrams that $\triangle$ *is associative.*

3) Let A, B be sets. Use associativity and other laws on $\triangle$, *which you should state, to simplify the following:*

$$(A \triangle B) \triangle (A \triangle (B \triangle A))$$

Your answer may involve A and/or B, but should have no occurrences of $\triangle$.

Exercise 23 *Say whether each of the following relations on* $\{1,2,3,4,5\}$ *is reflexive, symmetric or transitive (drawing directed graphs may help you).*

1) $R_1 = \{\langle 3,2\rangle, \langle 1,4\rangle, \langle 5,5\rangle, \langle 4,1\rangle\}$

2) $R_2 = \{\langle 2,2\rangle\}$

3) $R_3 = \{\langle 4,2\rangle, \langle 1,4\rangle, \langle 3,4\rangle, \langle 1,2\rangle, \langle 3,2\rangle\}$

4) $R_4 = \{\langle 1,1\rangle, \langle 2,2\rangle, \langle 3,3\rangle, \langle 4,4\rangle, \langle 5,5\rangle, \langle 1,5\rangle\}$

5) $R_5 = \{\langle 1,1\rangle, \langle 2,2\rangle, \langle 3,3\rangle, \langle 4,4\rangle, \langle 5,5\rangle, \langle 1,5\rangle, \langle 5,1\rangle\}$

Exercise 24 *1) Give matrix and directed graph representations of the following binary relation on* $\{1,2,3,4\}$:

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

2) Also give matrix and directed graph representations for $\overline{R}$ *and* R^{-1}.

Exercise 25 *Let R, S be binary relations on a set A.*

1) Show that $(R \cup S)^{-1} = R^{-1} \cup S^{-1}$.

2) Use (1) to deduce that $R \cup R^{-1}$ *is symmetric.*

3) Show that $(R \circ S)^{-1} = S^{-1} \circ R^{-1}$

4) Now suppose that $R \subseteq S$ *and S are symmetric. Show that* $R \cup R^{-1} \subseteq S$.

5) Use part (3) to show that R symmetric implies $R \circ R$ *is symmetric.*

Exercise 26 *Let R be a binary relation on A.*

1) Explain why if R is symmetric, also R^+ *is symmetric.*

2) Show that $(R \cup R^2)^+ = R^+$.

Exercise 27 *Let* $A = \{1,2,3,4\}$ *and consider the relation*

$$R = \{\langle 1,1\rangle, \langle 2,2\rangle, \langle 2,3\rangle, \langle 3,2\rangle, \langle 4,2\rangle, \langle 4,4\rangle\} \subseteq A^2$$

Is R reflexive, symmetric, or transitive? If not, illustrate your answer by listing the elements that are missing.

Exercise 28 Give examples of relations on $\{1,2,3,4\}$ having the following properties:

1) *reflexive, symmetric, not transitive.*
2) *reflexive, not symmetric, not transitive.*
3) *not reflexive, not symmetric, and transitive.*
4) *symmetric, transitive, not reflexive.*

Exercise 29 Let R and S be binary relations on A.

1) *Give specific relations R and S, such that R and S are transitive but $R \cup S$ is not.*
2) *Prove that if R and S are transitive then $R \cap S$ is transitive.*
3) *Give specific relations R and S, such that R and S are transitive but $R \circ S$ is not.*

Exercise 30 A relation R on a set A is called circular *if*

$$\forall a,b,c \in A\ (aRb \wedge bRc \Rightarrow cRa)$$

Prove that a relation R is an equivalence relation on A if and only if it is reflexive and circular on A.

Exercise 31 1) *Suppose $R = \{\langle 1,2\rangle, \langle 2,3\rangle, \langle 3,4\rangle, \langle 4,1\rangle\}$. Give R^+. Illustrate both R and its transitive closure R^+ as directed graphs.*

2) *Let R be the binary relation on the natural numbers defined by $x\,R\,y \overset{\Delta}{=} y = 2x$. Give R^+.*

Exercise 32 Find a binary relation R on a set A with $|A| = n$ such that $R^+ \neq R \cup R^2 \cup \ldots \cup R^{n-1}$.

3 Peano arithmetic

During the second half of the nineteenth century, mathematicians became aware of the fact that there was a need for more rigour in mathematics. In 1889 Peano defined a set of axioms for the natural numbers which have survived until today.

Definition 3.1 The set $I\!N$ is defined as the set satisfying:

1) $0 \in I\!N$.
2) If $n \in I\!N$, then $Succ(n) \in I\!N$ ($Succ$ is the successor function).
3) For all $n \in I\!N$, $Succ(n) \neq 0$.
4) For all n, $m \in I\!N$, if $Succ(n) = Succ(m)$, then $n = m$.
5) Let V be a set such that $0 \in V$ and, for all $n \in I\!N$, if $n \in V$ then $Succ(n) \in V$, then $I\!N \subseteq V$.

By clause (*4*), *Succ* is an injection (see Definition 4.10). Clause (*5*) is sometimes called the *axiom* (or *principle*) *of induction* (we will come back to this in Section 9). It corresponds to demanding that $I\!N$ is the *smallest set* produced by this definition, so there cannot be an element in $I\!N$ that is both not the same as 0 and not the successor of an element in $I\!N$.

Peano's definition presents the natural numbers in a unary representation: we would define $1 \overset{\Delta}{=} Succ(0)$, $2 \overset{\Delta}{=} Succ(Succ(0))$ (which then is also $Succ(1)$), etc.; informally, a natural number n is the result of applying the constructor *Succ* n-times to 0. Moreover, the set $I\!N$ is infinitely large and contains 0, $Succ(0)$, $Succ(Succ(0))$, $Succ(Succ(Succ(0)))$, … that are all different.

Example 3.2 To illustrate that this is a working definition, we can now define the operations over numbers, like addition and multiplication:

$$\begin{array}{rcl} Add\ (0, \quad\quad\quad n) & = & n \\ Add\ (Succ(m), n) & = & Succ(Add(m,n)) \\ \\ Mult\ (0, \quad\quad\quad n) & = & 0 \\ Mult\ (Succ(m), n) & = & Add(Mult(m,n),n) \end{array}$$

etc. Notice that this uses Haskell-style pattern matching. We can show that $3+4=7$, as expected.

$$\begin{array}{rcl} Add(3,4) & \triangleq & Add(Succ(Succ(Succ(0))),\ Succ(Succ(Succ(Succ(0))))) \\ & = & Succ(Add(Succ(Succ(0)),\ Succ(Succ(Succ(Succ(0)))))) \\ & = & Succ(Succ(Add(Succ(0,)\ Succ(Succ(Succ(Succ(0))))))) \\ & = & Succ(Succ(Succ(Add(0,\ Succ(Succ(Succ(Succ(0)))))))) \\ & = & Succ(Succ(Succ(Succ(Succ(Succ(Succ(0))))))) \qquad \triangleq\ 7 \end{array}$$

We can also define the normal partial order relation on $I\!N$.

Definition 3.3 We can define a smaller-than relation $<_1$ on numbers through:

$$<_1\ \triangleq\ \{\langle n, Succ(n)\rangle \mid n \in I\!N\}$$

so this relation contains only the pairs of consecutive numbers. Now the 'normal' smaller-than relation on numbers is defined as the transitive closure of this relation.

$$<_{I\!N}\ \triangleq\ (<_1)^+$$

We can define other infinite sets from $I\!N$:

Definition 3.4 Having the set $I\!N$ at hand, we can now define $\mathbb{Z}$ by:

$$\begin{array}{rcl} \mathbb{Z} & \triangleq & I\!N \cup \{-n \mid n \in I\!N\} \\ =_{\mathbb{Z}} & \triangleq & \{\langle n,m\rangle \mid n =_{I\!N} m\} \cup \{\langle -n,-m\rangle \mid n =_{I\!N} m\} \cup \{\langle -0,0\rangle\} \end{array}$$

On the other hand, we now can define $\mathbb{Q}$ via:

$$\begin{array}{rcl} \mathbb{Q} & \triangleq & \mathbb{Z} \times I\!N \\ =_{\mathbb{Q}} & \triangleq & \{\langle\langle n_1,m_1\rangle,\langle n_2,m_2\rangle\rangle \in (\mathbb{Z}\times I\!N)^2 \mid n_1 \times m_2 =_{I\!N} n_2 \times m_1\} \end{array}$$

Notice that $=_{\mathbb{Q}}$, by Example 2.16, is an equivalence relation.

Defining $I\!R$ is a bit more work.

But we can ask an even more fundamental question: do natural numbers really *exist*? Accepting that sets are 'real', in set theory we can define an interpretation of the natural numbers recursively by:

$$\begin{array}{rcl} 0 & \triangleq & \emptyset \\ Succ(n) & \triangleq & n \cup \{n\} \end{array}$$

Then

$$n\ =\ n-1 \cup \{n-1\}\ =\ n-2 \cup \{n-2\} \cup \{n-1\} \cdots\ =\ \{0,1,\ldots,n-2,n-1\}$$

for each natural number n. For example,

$$\begin{array}{rcccccccc} 0 & = & \emptyset \\ 1 & = & 0 \cup \{0\} & = & \emptyset \cup \{\emptyset\} & = & \{\emptyset\} & = & \{0\} \\ 2 & = & 1 \cup \{1\} & = & \{\emptyset\} \cup \{\{\emptyset\}\} & = & \{\emptyset,\{\emptyset\}\} & = & \{0,1\}^{11} \\ 3 & = & 2 \cup \{2\} & = & \{\emptyset,\{\emptyset\}\} \cup \{\{\emptyset,\{\emptyset\}\}\} & = & \{\emptyset,\{\emptyset\},\{\emptyset,\{\emptyset\}\}\} & = & \{0,1,2\} \\ & \vdots \end{array}$$

This gives a *model* of Peano arithmetic.

Although numbers can be defined formally, the definition is not complete. In fact, Kurt Gödel has shown that in any effectively generated axiomatisation of number theory, there will be a statement, expressed in number theory, which cannot be proven or disproven in that system; this is his famous *incompleteness result*. So, no matter how we define the numbers, there will be statements that escape our reasoning. So formal set theory or any other effectively generated formal system can never express correctly, completely, what is a number.

Exercises

Exercise 33 Show that, for all $n \in I\!N$, $Add(n, Succ(0)) = Succ(n)$.

[11] Sometimes $\mathbb{2}$ is written for the set $\{0,1\}$, as in $\mathbb{2}^V$ for the set of characteristic functions for subsets of V (see Definition 4.6).

4 Functions

In this section we will formalise the notion of *mathematical function* as a special kind of relation, giving the basic definitions, ways of constructing functions, and properties for reasoning about functions. We will give the definition of *computable functions*, a notion that underpins computer science.

4.1 Introducing Functions

Definition 4.1 (FUNCTIONS) A function f from a set A to a set B, written $f : A \to B$, is a relation $f \subseteq A \times B$ such that every element of A is related to *one* element of B; more formally, it is a relation which satisfies the following properties:

$$\forall b_1, b_2 \in B \; (\, \exists a \in A \; (\langle a, b_1 \rangle \in f \wedge \langle a, b_2 \rangle \in f \,) \Rightarrow b_1 = b_2 \,)$$
$$\forall a \in A \; \exists b \in B \; (\, \langle a, b \rangle \in f \,)$$

1) The set A is called the *domain* (or *source type*) *of* f, and B is the *co-domain* (or *range*, or *target type*) of f.

2) If $a \in A$, then $f(a)$ (f *applied to* the *argument* a) denotes the unique $b \in B$ such that $\langle a, b \rangle \in f$, and f is said to *map* a to b. We also say that b is the *image of* a *under* f, or that a is a *pre-image* (or *inverse image*) of b under f. Note that elements of the domain always have a single image, but elements of the co-domain can have more than one pre-image or may have none.

3) If the domain A is the n-ary product $A_1 \times \cdots \times A_n$, we normally write $f(a_1, \ldots, a_n)$ instead of $f(\langle a_1, \ldots, a_n \rangle)$; we say that f is then a function of n arguments or an n-ary function.

4) We write B^A for the set of all functions from A to B.[12]

We see $f : A \to B$ as shorthand for $f \in B^A$, and are allowed to write $\exists f : A \to B \; (\cdots)$ for $\exists f \in B^A \; (\cdots)$ and $\forall f : A \to B \; (\cdots)$ for $\forall f \in B^A \; (\cdots)$, since we are essentially quantifying over the set B^A.

Notice that we can now state that f is a function if it satisfies:

$$\forall a \in A, b_1, b_2 \in B \; (f(a) = b_1 \wedge f(a) = b_2 \Rightarrow b_1 = b_2)$$
$$\forall a \in A \; \exists b \in B \; (f(a) = b)$$

We define equality on functions.

Definition 4.2 Let $f : A \to B$ and $g : A \to B$. Then $f =_{A \to B} g \;\triangleq\; \forall x \in A \; (f(x) =_B g(x))$.

So two functions are equal when they have the same domain and return the same value for every element of that domain.

Definition 4.3 Let $f : A \to B$. For any $X \subseteq A$, we define the image of X under f to be

$$f[X] \;\triangleq\; \{ f(a) \in B \mid a \in X \}$$

The set $f[A] \subseteq B$ of all images of f is called the *image set* of f.

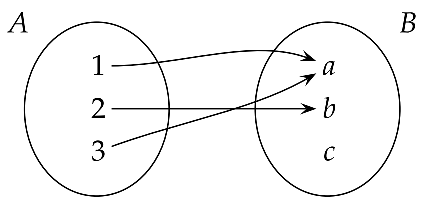
Notice that, in general, $f[A] \neq B$.

We explore some examples.

Example 4.4 Since functions are defined as binary relations, we can use the representations given in Section 2 for relations to describe functions. When the domain and co-domain are finite and not too large, a useful representation is the *diagram* representation.

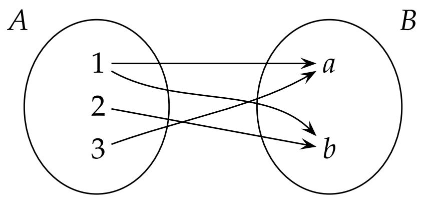
1) Let $A = \{1,2,3\}$ and $B = \{a,b,c\}$. Let $f : A \to B$ (notice that then $f \subseteq A \times B$) be defined by $f = \{ \langle 1,a \rangle, \langle 2,b \rangle, \langle 3,a \rangle \}$. Then f is a function, as is clear from the diagram:

[12] This notation is justified by Proposition 4.5.



Then $f[\{1,2,3\}] = \{a,b\}$; $f[\{1,3\}]$, the image of $\{1,3\}$ under f, is $\{a\}$.

2) Let $A = \{1,2,3\}$, $B = \{a,b\}$, and $f \subseteq A \times B$ be defined by $f = \{\langle 1,a\rangle, \langle 1,b\rangle, \langle 2,b\rangle, \langle 3,a\rangle\}$. This f is not a function, since one element of A is related to two elements in B as is evident from the diagram:



3) The following are examples of functions with infinite domains and codomains:
a) the function $f : I\!N^2 \to I\!N$ defined by $f(x,y) = x + y$;
b) the function $f : I\!N \to I\!N$ defined by $f(x) = x^2$;
c) the function $f : I\!R \to I\!R$ defined by $f(x) = x + 3$.

4) The binary relation R on the reals defined by $x\,R\,y$ when $x = y^2$ is not a function, since for example 4 relates to both 2 and -2.

In the finite case, we can give an exact measure for the set of functions from A to B.

Proposition 4.5 If $|A| = m$ *and* $|B| = n$*, then* $|B^A| = n^m$.

Proof: For each element of A, there are m independent ways of mapping it to an element of B. We can view this as establishing how many numbers we can represent with m positions in base n: the answer to this is n^m. ■

Definition 4.6 (CHARACTERISTIC FUNCTION) *1*) Let A be a set. The *characteristic function of* $B \subseteq A$ is the function $\chi_B : A \to \{0,1\}$ defined as:

$$\chi_B(a) \;=\; \begin{cases} 1 & (a \in B) \\ 0 & (a \in A \setminus B) \end{cases}$$

2) The *characteristic function of the relation* $R \subseteq A_1 \times \cdots \times A_n$ is the function $\chi_R : A_1 \times \cdots \times A_n \to \{0,1\}$ defined as:

$$\chi_R(a_1,\ldots,a_n) \;=\; \begin{cases} 1 & (\langle a_1,\ldots,a_n\rangle \in R) \\ 0 & (\langle a_1,\ldots,a_n\rangle \notin R) \end{cases}$$

In the literature, the characteristic function of a set B is also written as 1_B, I_B, or f_B; notice that the 'surrounding set' A is implicit in that we assume we know what B is a subset of.

4.2 Partial Functions

We have defined functions to be total (i.e. to have a value for every argument in the domain), following usual mathematical practice. A *partial* function is a function which need not be defined on every member of its domain. It therefore assigns to each element of its domain *at most* one element of the range.

Definition 4.7 A *partial function* f from a set A to a set B is a relation $f \subseteq A \times B$ such that just some elements of A are related to unique elements of B; more formally, it is a relation which satisfies only the first clause of Definition 4.1:

$$\forall a \in A, b_1, b_2 \in B\; (\langle a,b_1\rangle \in f \wedge \langle a,b_2\rangle \in f \Rightarrow b_1 = b_2)$$

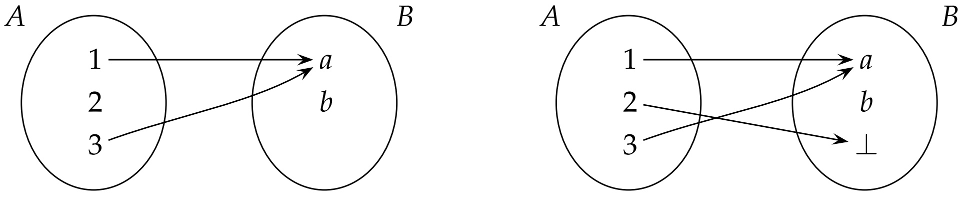
A function that also satisfies the second clause

$$\forall a \in A\; \exists b \in B\; (\langle a,b\rangle \in f)$$

so has an image for each element of A, is then called *total*.

The partial function f is regarded as *undefined* on those elements which do not have an image under f. It is sometimes convenient to refer to this undefined value explicitly as $\perp$ (bottom); a partial function from A to B is straightforwardly extended to a (total) function from A to $B \cup \{\perp\}$.

Example 4.8 The relation $R = \{\langle 1,a\rangle, \langle 3,a\rangle\}$ is a partial function, shown in the diagram on the left. We can see that not every element in A maps to an element in B. It is extended to a total function as in the diagram on the right:



Example 4.9 The binary relation R on $\mathbb{R}$ defined by $R \triangleq \{\langle x,y\rangle \in \mathbb{R}^2 \mid \sqrt{x} = y\}$ is a partial function (it is not defined when x is negative).

From now on, when we use the term ‘function’, we implicitly assume that we are dealing with *total* functions, so will only treat partial functions when explicitly stating so.

4.3 Properties of Functions

Recall that we highlighted certain properties of relations, such as reflexivity, symmetry and transitivity. Since functions are special relations, we can define the identity, composition and inverse relations of functions. The composition of two functions will turn out to always be a function. In contrast, we shall see that the inverse relation of a function need not necessarily be a function.

Definition 4.10 (PROPERTIES OF FUNCTIONS) Let $f : A \to B$ be a function. We define the following properties on f:

(f *is* onto *(surjective)*): every element of B is in the image of f; that is:

$$\forall b \in B\ \exists a \in A\ (f(a) = b)$$

(f *is* one-to-one *(injective)*): for each $b \in B$ there exists at most one $a \in A$ with $f(a) = b$; that is:

$$\forall a,a' \in A\ (f(a) = f(a') \Rightarrow a = a')$$

(f *is* bijective): f is both *one-to-one* and *onto*.

By contraposition, we can formulate f being injective also as $\forall a,a' \in A\ (a \neq a' \Rightarrow\ (f(a) \neq f(a')))$.

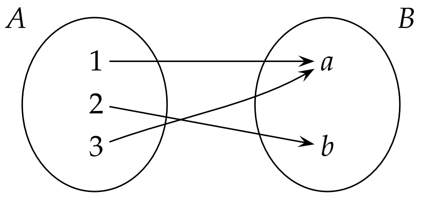
Notice that the criterion of a *one-to-one* function is (by contra-position[13]) equivalent to

$$\forall a,a' \in A\ (a \neq a' \Rightarrow f(a) \neq f(a'))$$

and so a *one-to-one* function never repeats values.

The previous definition is stated for total functions only; if we would like to extend the properties to partial functions, we have to make them total, by either restricting the domain (eliminating all the elements for which the function is not defined), or extending the range with $\perp$.

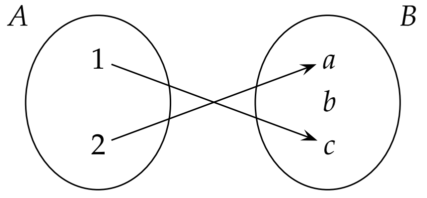
Example 4.11 1) Let $A = \{1,2,3\}$ and $B = \{a,b\}$. The function $f = \{\langle 1,a\rangle, \langle 2,b\rangle, \langle 3,a\rangle\}$ is *onto*, but not *one-to-one*, as is immediate from its diagram:



[13] Contra-position states that $P \to Q \Leftrightarrow \neg Q \to \neg P$, and $\neg Q \to \neg P$ is the contra-positive of $P \to Q$.

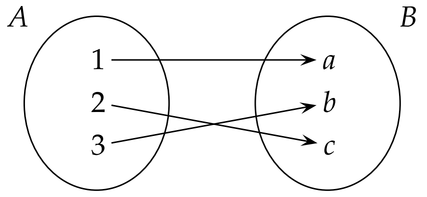
It is not possible to define a *one-to-one* function from A to B, as there are too many elements in A for them to map uniquely to B.

2) Let $A = \{1,2\}$ and $B = \{a,b,c\}$. The function $f = \{\langle 1,c\rangle, \langle 2,a\rangle\}$ is *one-to-one*, but not *onto*:



It is not possible to define an *onto* function from A to B in this case, as there are not enough elements in A to map to all the elements of B.

3) Let $A = \{1,2,3\}$ and $B = \{a,b,c\}$. The function $f = \{\langle 1,a\rangle, \langle 2,c\rangle, \langle 3,b\rangle\}$ is bijective:



4) The function $f : I\!N^2 \to I\!N$ defined by $f(x,y) = x+y$ is *onto* but not *one-to-one*. To prove that f is *onto*, take an arbitrary $n \in I\!N$. We must find $\langle m_1, m_2\rangle \in I\!N^2$ such that $f(m_1,m_2) = n$. This is easy since $f(n,0) = n+0 = n$. To show that f is not *one-to-one*, we need to produce a counter example. In other words, we must find $\langle m_1,m_2\rangle$, $\langle n_1,n_2\rangle$ such that $\langle m_1,m_2\rangle \neq \langle n_1,n_2\rangle$, but $f(m_1,m_2) = f(n_1,n_2)$. There are many possibilities, such as $\langle 1,0\rangle$ and $\langle 0,1\rangle$. In fact, since $+$ is commutative, $\langle m,n\rangle$, $\langle n,m\rangle$ is a counter example for any m and n such that $m \neq n$.

5) The function $f : I\!N \to I\!N$ defined by $f(x) = x^2$ is *one-to-one*, but the similar function f on integers $Z\!\!\!Z$ is not. The function f on $Z\!\!\!Z$ defined by $f(x) = x+1$ is surjective, but the similar function on $I\!N$ is not.

6) The function $f : I\!R \to I\!R$ given by $f(x) = 4x+3$ is a bijective function. To prove that f is *one-to-one*, suppose that $f(r_1) = f(r_2)$, which means that $4r_1+3 = 4r_2+3$. It follows that $4r_1 = 4r_2$, and hence $r_1 = r_2$. To prove that f is *onto*, let r be an arbitrary real number. We have $f((r-3)/4) = 4((r-3)/4)+3 = (r-3)+3 = r$ by definition of f, and hence f is *onto*. Since f is both *one-to-one* and *onto*, it is bijective.

   Notice that the restriction of f to $I\!N$ is *one-to-one* but not *onto*, since 2 is not in $f[I\!N]$.

We have seen examples that we cannot define a bijection between finite sets A and B when $|A| > |B|$; any total function would need to map two elements of A to the same element of B. This corresponds to the pigeonhole principle, which we can rephrase in our formal language of functions:

Let $f : A \to B$ be a function, where A and B are finite. If $|A| > |B|$, then f is not injective.

Recall Example 4.11. We stated that is not possible to define a *one-to-one* function from $A = \{1,2,3\}$ to $B = \{a,b\}$, since A is too big, and the pigeonhole principle confirms this.

Proposition 4.12 Let A and B be finite sets, let $f : A \to B$ and let $X \subseteq A$. Then $|f[X]| \leq |X|$.

Proof: This property is intuitively clear, and can be proved by appealing to the pigeonhole principle.

Assume towards a contradiction that $|f[X]| > |X|$. Define a place function $p : f[X] \to X$ by

Let $b \in f[X]$; take $p(b) = a$ for some $a \in X$ such that $f(a) = b$.

It does not matter which a we choose; there will be such an a by definition of $f[X]$. We are placing the members of $f[X]$ in the pigeonhole X. By the pigeonhole principle, some pigeonholes has at least two occupants. In other words, there exists some $a \in X$ and $b, b' \in f[X]$ with $b \neq b'$ and $p(b) = p(b') = a$. But then $f(a) = b$ and $f(a) = b'$, which cannot happen as f is a function. Therefore our assumption is false, and $\neg(|f[X]| > |X|)$, so $|f[X]| \leq |X|$. ■

Proposition 4.13 Let A and B be finite sets, and let $f : A \to B$.

1) If f is one-to-one*, then $|A| \leq |B|$.*
2) If f is onto*, then $|A| \geq |B|$.*
3) If f is a bijection, then $|A| = |B|$.

Proof: Part (*1*) is the contra-positive of the pigeonhole principle formulated for functions. For (*2*), notice that if f is *onto* then $f[A] = B$, so that in particular $|f[A]| = |B|$. Also, $|A| \geq |f[A]|$ by Proposition 4.12. Therefore $|A| \geq |B|$ as required. Finally, part (*3*) follows from parts (*1*) and (*2*). ■

Cantor has shown the following property, that relates injections and surjections with bijections:

Theorem 4.14 ((DUAL) CANTOR-BERNSTEIN THEOREM) *If there exists functions $f : A \to B$ and $g : B \to A$, both injective or both surjective, then there exists a bijection $h : A \to B$.*

We will not go into the details of the proof for this property here, but it will be useful in Section 4.5.

4.4 Operations on Functions

Since functions are relations, we can define identity, composition, and inverse of functions. We shall see that the inverse relation of a function need not necessarily be a function.

Definition 4.15 (FUNCTION COMPOSITION) Let A, B, and C be arbitrary sets, and let $f : A \to B$ and $g : B \to C$ be arbitrary functions. The composition of f with g, written $g \circ f : A \to C$, is a function defined by

$$g \circ f(a) \;\triangleq\; g(f(a))$$

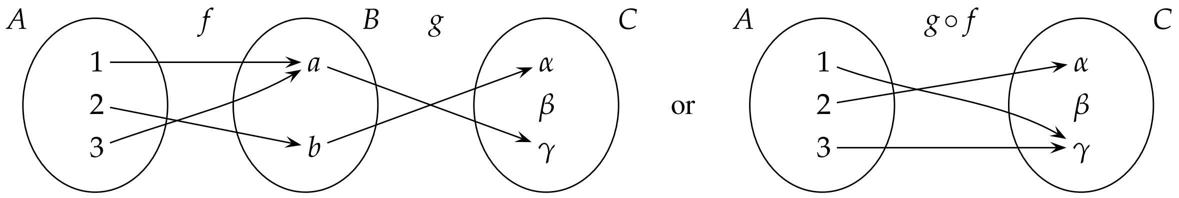
for every element $a \in A$. In other words,

$$g \circ f(a) = c \Leftrightarrow \exists b \in B\ (f(a) = b \wedge g(b) = c)$$

Notice that composition of relations is different from functional composition. Functional composition $g \circ f$ reads *f followed by g* (or rather *g after f*), whereas the relational composition $R \circ S$ reads *R followed by S*. The main reason for the difference is that $g \circ f$ is a prefix notation, whereas $R \circ S$ is infix. Likewise, if we view functions f and g as relations F and G and we have $g(fa) = c$, then $a\, F \circ G\, c$, so by definition of relation composition $\exists b\ (a\, F\, b \wedge b\, G\, c)$.

It is easy to check that $g \circ f$ is indeed a function. Notice that the co-domain of f must be the same as the domain of g for the composition to be well defined.

Example 4.16 Let $A = \{1,2,3\}$, $B = \{a,b,c\}$, $C = \{\alpha,\beta,\gamma\}$, $f : A \to B = \{\langle 1,a\rangle, \langle 2,b\rangle, \langle 3,a\rangle\}$ and $g : B \to C = \langle a,\gamma\rangle, \langle b,\alpha\rangle$. Then $g \circ f : A \to C = \{\langle 1,\gamma\rangle, \langle 2,\alpha\rangle, \langle 3,\gamma\rangle\}$:



Proposition 4.17 (ASSOCIATIVITY OF FUNCTION COMPOSITION) *Let $f : A \to B$, $g : B \to C$, and $h : C \to D$ be arbitrary functions. Then $h \circ (g \circ f) = (h \circ g) \circ f$.*

Proof: This result is easily established from the definition of functional composition. Take an arbitrary element $a \in A$. Then

$$(h \circ (g \circ f))(a) \;=\; h((g \circ f)(a)) \;=\; h(g(f(a))) \;=\; (h \circ g)(f(a)) \;=\; ((h \circ g) \circ f)(a) \quad ■$$

The proof can be illustrated by this diagram:

$$\begin{tikzcd}
A \arrow[r, "f"] \arrow[dr, "g\circ f"'] & B \arrow[d, "g"'] \arrow[dr, "h\circ g"] & \\
& C \arrow[r, "h"'] & D
\end{tikzcd}$$

Each of the two triangles ABC, BDC 'commutes': that is, the result is the same whether one follows f followed by g or $g\circ f$, and whether one does g followed by h or $h\circ g$. The parallelogram $ABCD$ therefore commutes, which means that the result holds.

Proposition 4.18 If $f : A \to B$ and $g : B \to C$ are bijections, then so is $g\circ f$.

Proof: It is enough to show that:

1) if f, g are *onto* then so is $g\circ f$;

2) if f, g are *one-to-one* then so is $g\circ f$.

For point (*1*), assume f and g are *onto*. We need to show that for every element c in C there exists an element a in A such that $g\circ f(a) = c$. So, let c be an arbitrary element of C; since g is *onto*, we can find an element $b\in B$ such that $g(b) = c$. Since f is *onto*, we can also find an element $a\in A$ such that $f(a) = b$. But then $g\circ f(a) = g(f(a)) = g(b) = c$, and hence $g\circ f$ is *onto*.

For point (*2*) , assume f and g are *one-to-one*. We need to show that for every two elements a_1 and a_2 in A, if $g\circ f(a_1) = g\circ f(a_2)$ then $a_1 = a_2$. So, let a_1, a_2 be arbitrary elements of A, and suppose $g\circ f(a_1) = g\circ f(a_2)$. Then $g(f(a_1)) = g(f(a_2))$ by definition of $g\circ f$. Since g is *one-to-one*, it follows that $f(a_1) = f(a_2)$. Since f is also *one-to-one*, it follows that $a_1 = a_2$, so $g\circ f$ is *one-to-one*. ■

Definition 4.19 (IDENTITY FUNCTION) Let A be a set. The identity function on A, denoted $id_A : A \to A$, is defined by $id_A(a) = a$.

We shall now define the inverse function. We cannot just define it using the definition of inverse relations, as we do not always get a function this way. For example, the inverse relation of the function $f : \{1,2\} \to \{1\}$ defined by $f(1) = f(2) = 1$ is not a function. However, we shall see that when inverse functions exist, they correspond to the inverse relation.

Definition 4.20 (INVERSE FUNCTION) let $f : A \to B$ be an arbitrary function. The function $g : B \to A$ is an *inverse of f* whenever

$$\forall a\in A\ (g(f(a)) = a)$$
$$\forall b\in B\ (f(g(b)) = b)$$

Another way of stating the same property is that $g\circ f = id_A$ and $f\circ g = id_B$.

Example 4.21 Let $A = \{a,b,c\}$, $B = \{1,2,3\}$, $f = \{\langle a,1\rangle,\langle b,3\rangle,\langle c,2\rangle\}$ and $g = \langle 1,a\rangle,\langle 2,c\rangle,\langle 3,b\rangle$. Then g is an inverse of f.

Proposition 4.22 Let $f : A \to B$ be a bijection, and define $f^{-1} : B \to A$ by $f^{-1}(b) = a$ whenever $f(a) = b$. In this case, the relation f^{-1} is a well-defined function, and is an inverse of f (in fact, the inverse in view of the next proposition).

Proof: Let $b\in B$ be arbitrary. Since f is *onto*, there exists an a such that $f(a) = b$. Since f is *one-to-one*, this a is unique. This means that f^{-1} is a function. By definition, it satisfies the conditions for being an inverse of f. ■

Proposition 4.23 Let $f : A \to B$. If f has an inverse g, then f must be a bijection and the inverse is unique (and is the f^{-1} given in Proposition 4.22).

Proof: Let g be the inverse of f. To show that f is *onto*, take an arbitrary element b of B. Since $f(g(b)) = b$, it follows that b must be in the image of f. To show that f is *one-to-one*, let a_1, a_2 be arbitrary elements of A. Suppose $f(a_1) = f(a_2)$, which implies that $g(f(a_1)) = g(f(a_2))$. Since $g\circ f = id_A$, it follows that $a_1 = a_2$.

To show that the inverse is unique, assume that g, g' are both inverses of f. We will show that $g = g'$. Let b be an arbitrary element of B. Then $f(g(b)) = f(g'(b))$ since g and g' are both inverses of f. Hence $g(b) = g'(b)$ since f is *one-to-one*. Therefore, for all b, $g(b) = g'(b)$, so $g = g'$ follows by Definition 4.2. ■

In view of the preceding proposition, one way of showing that a function is a bijection is to show that it has an inverse. Furthermore, if f is a bijection with inverse f^{-1}, then f^{-1} has an inverse, namely f, and so f^{-1} is also a bijection.

Example 4.24 1) Take $f : I\!R \to I\!R$ defined by $f(x) = x^3$. The inverse is $f^{-1}(x) = \sqrt[3]{x}$.

2) Take the function $f : I\!N \to I\!N$ defined by

$$f(x) = \begin{cases} x+1 & (x \textit{ even}) \\ x-1 & (x \textit{ odd}) \end{cases}$$

It is easy to check that $f \circ f(x) = x$, considering the cases when x is odd and even separately. Therefore f is its own inverse, and we can conclude that it is a bijection.

4.5 Cardinality of Sets

We are now in a position to compare the size of infinite sets, using a natural relation between sets defined using bijective functions.

Definition 4.25 We define the relation $\approx$ on sets as:[14]

$$A \approx B \;\triangleq\; \exists f : A \to B \;(f \textit{ is a bijection}).$$

We can now restate Theorem 4.14 as follows:

Theorem 4.26 *If there exists functions $f : A \to B$ and $g : B \to A$, both injective or both surjective, then $A \approx B$.*

We can use this result to show the following:

Example 4.27 $\{blue, red\} \approx \{cat, dog\}$; $I\!N \approx \{n \in I\!N \mid n \textit{ is even}\}$; $I\!N \approx \{n \in I\!N \mid n \textit{ is prime}\}$.

We will show that $I\!N \approx Z\!\!Z$ in Example 6.3, $I\!N \approx I\!N^2$ in Example 4.33, $I\!N \approx \mathbb{Q}$ in Example 6.4, $I\!N \approx \{V \subseteq I\!N \mid \exists n \in I\!N\, (|V| = n)\}$ in Example 6.5, $I\!N \not\approx \wp I\!N$ in Example 6.6, and $I\!N \not\approx I\!R$ in Example 6.7.

Proposition 4.28 The relation $\approx$ on sets is an equivalence relation.

Proof: We need to show that $\approx$ is reflexive, symmetric, and transitive.

- The relation $\approx$ is reflexive, since $id_A : A \to A$ is clearly a bijection.
- To show that it is symmetric, observe that by definition $A \approx B$ implies that there exists a bijection $f : A \to B$. By Proposition 4.22, it follows that f has an inverse $f^{-1} : B \to A$ which is also a bijection, hence $B \approx A$.
- That the relation $\approx$ is transitive follows from Proposition 4.18. ■

* *Example 4.29* Let A, B, and C be arbitrary sets, and consider the products $(A \times B) \times C$ and $A \times (B \times C)$. Remember that we observed in Section 1.4 that these sets are not in general equal.

There exists however a natural bijection $f : (A \times B) \times C \to A \times (B \times C)$ such that $f(\langle a,b \rangle, c) = \langle a, \langle b,c \rangle\rangle$. Similarly, there exists a bijection $g : A \times (B \times C) \to (A \times B) \times C$ such that $g(a, \langle b,c \rangle) = \langle\langle a,b \rangle, c\rangle$. In particular, we define

[14] In the literature, often the symbol $\sim$ is used rather than $\approx$. In these notes, to avoid confusion, we reserve that symbol for equivalence relations.

$$\begin{aligned} \mathit{Left}(x,y) &= x \\ \mathit{Right}(x,y) &= y \\ f(x,y) &= \langle \mathit{Left}(x), \langle \mathit{Right}(x), y\rangle\rangle \\ g(x,y) &= \langle\langle x, \mathit{Left}(y)\rangle, \mathit{Right}(y)\rangle \end{aligned}$$

It is not difficult to show that $g \circ f = id_{A\times(B\times C)}$ and $f \circ g = id_{(A\times B)\times C}$; using Proposition 4.23 it follows that f is a bijection.

Example 4.30 Consider the set Even of even natural numbers. There exists a bijection between Even and $I\!N$, $f : I\!N \to \mathsf{Even}$ given by $f(n) = 2n$. Notice that not all functions from Even to $I\!N$ are bijections: for example, the function $g : \mathsf{Even} \to I\!N$ given by $g(n) = n$ is *one-to-one* but not *onto*.

To show that $\mathsf{Even} \approx I\!N$ it is enough to show the existence of one such bijection.

The following gives another explanation for the notation for the set of functions from A to B.

Example 4.31 Let A be such that $|A| = n$, and take the set $\{\chi_B : A \to \{0,1\} \mid B \subseteq A\}$. Since each characteristic function for a subset of A is a function from A to $\{0,1\}$, we have

$$\{\chi_B : A \to \{0,1\} \mid B \subseteq A\} = \{f \mid f : A \to \{0,1\}\}$$

By definition, we can write the latter set as $\{0,1\}^A$, or as $\mathbf{2}^A$.[15] Now

$$\wp A = \{B \mid B \subseteq A\} \approx \{\chi_B : A \to \{0,1\} \mid B \subseteq A\} = \{f : A \to \{0,1\}\} = \mathbf{2}^A$$

and $|\wp A| = \mathbf{2}^n = \mathbf{2}^{|A|}$.

Recall that the cardinality of a finite set is the number of elements in that set. Consider a finite set A with cardinality n. Then there exists a counting bijection

$$c_A : \{1,2,\ldots,n\} \to A$$

This function should be familiar, in that we often enumerate the elements in A by $a_1,\ldots,a_n$. Now let A and B be two finite sets. If A and B have the same number of elements, then we can define a bijection $f : A \to B$ by

$$f(a) = (c_B \circ {c_A}^{-1})\,(a)$$

Thus, two finite sets have the same number of elements if and only if there exists a bijection between them.

We extend this observation to compare the size of *infinite* sets.

Definition 4.32 (CARDINALITY) Given two (arbitrary) sets A and B, we say that A has the *same cardinality* as B, written $|A| = |B|$, whenever there exists a bijection between A and B, so when $A \approx B$.

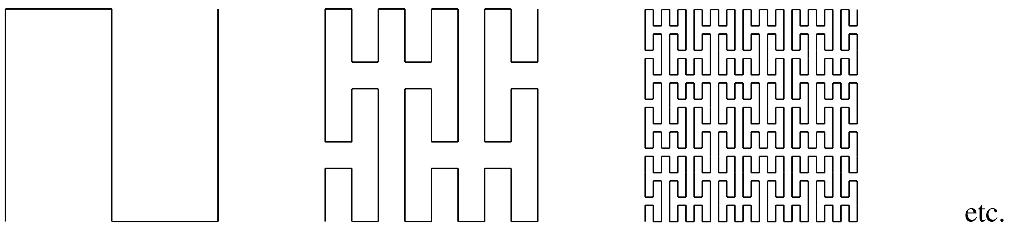
We will give some examples of sets that have the same cardinality.

Example 4.33 ($I\!N \approx I\!N^2$) We can build a bijection from $I\!N$ to $I\!N^2$ as illustrated by the following diagram:

$$\begin{tikzcd}
\langle 0,0\rangle \arrow[r] & \langle 0,1\rangle \arrow[dl] & \langle 0,2\rangle \arrow[r] & \langle 0,3\rangle \arrow[dl] & \langle 0,4\rangle \arrow[r] & \langle 0,5\rangle \arrow[dl] & \cdots \\
\langle 1,0\rangle \arrow[d, bend right] & \langle 1,1\rangle \arrow[ur] & \langle 1,2\rangle \arrow[dl] & \langle 1,3\rangle \arrow[ur] & \langle 1,4\rangle \arrow[dl] & \langle 1,5\rangle & \cdots \\
\langle 2,0\rangle \arrow[ur] & \langle 2,1\rangle \arrow[dl] & \langle 2,2\rangle \arrow[ur] & \langle 2,3\rangle \arrow[dl] & \langle 2,4\rangle & \langle 2,5\rangle & \cdots \\
\langle 3,0\rangle \arrow[d, bend right] & \langle 3,1\rangle \arrow[ur] & \langle 3,2\rangle \arrow[dl] & \langle 3,3\rangle & \langle 3,4\rangle & \langle 3,5\rangle & \cdots \\
\langle 4,0\rangle \arrow[ur] & \langle 4,1\rangle \arrow[dl] & \langle 4,2\rangle & \langle 4,3\rangle & \langle 4,4\rangle & \langle 4,5\rangle & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots &
\end{tikzcd}$$

Example 4.34 ($I\!R \approx I\!R^2$) We show that $I\!R \approx I\!R^2$ by first using Peano's solution to build a surjection from $(0,1)$ (the open interval of reals between 0 and 1) to $(0,1)^2$ in steps as illustrated by the following diagrams:

[15] Remember that in Peano arithmetic $2 = \{\emptyset, \{\emptyset\}\} = \{0,1\}$



In the limit, this will build a surjection onto $(0,1)^2$; the idea behind the proof is that for every element of $(0,1)^2$, and any circle cast around that point, there will be an incarnation of the construction that cuts through that circle. We also know that $g(x,y) = x$ is a surjection from $(0,1)^2$ to $(0,1)$; then by Theorem 4.26, we get $(0,1)^2 \approx (0,1)$.

We also have $(0,1) \approx I\!R$ via the bijection

$$f(x) = \tan(\pi \times (x - 1/2))$$

and thereby $I\!R \approx I\!R^2$ and also $\mathbb{C} \approx I\!R$.

Using his famous *diagonalisation argument*, Cantor showed in 1891 that no set can have the same cardinality as its power set. This result is clear for the case that A is finite, since we have seen that if $|A| = n$, then $|\wp A| = 2^n$, and $n \neq 2^n$ for all $n \in I\!N$.

The proof is by contradiction: it assumes a surjection f exists, and using f constructs a set that is *outside* the range of f, which therefore cannot be a surjection.

Theorem 4.35 (Cantor's theorem) *For any set A, $A \not\approx \wp A$.*

Proof: Assume that there exists a function $f : A \to \wp A$. Notice that since f maps elements of A to subsets of A, every element a of A is either an element of the set $f(a)$, or not. Define

$$B = \{a \in A \mid a \notin f(a)\},$$

which is a well-defined subset of A. Now assume f is surjective; then there must exist $b \in A$ such that $f(b) = B$. Then either:

$(b \in B)$: then $b \notin f(b) = B$, so $b \notin B$; or

$(b \notin B)$: then $b \notin f(b)$, so $b \in B$.

This is impossible. So there is no b such that $f(b) = B$, so f is not surjective. This is a contradiction, so there exists no surjection $f : A \to \wp A$, so certainly no bijection. ∎

We recognise Russell's paradox here. The difference is that now B *is* a set, and that the contradiction lies in the fact that B is not in the image of f, as was assumed.

Notice that the above result is formulated for *any* set, not just finite or infinite sets.

An illustration of this result is that $I\!N \not\approx \wp I\!N$; we will come back to this in the next section. So $I\!N$ and $\wp I\!N$ have a different cardinality (size of the set), as do $\wp(\wp I\!N)$, $\wp(\wp(\wp I\!N))$, etc. In fact, we have an infinite hierarchy of measures of infinity, each much larger than the other. This concept is widely accepted at the moment, but was quite controversial when introduced by Cantor.

Exercises

Exercise 34 Let $A = \{1,2\}$ and $B = \{a,b,c\}$. List the elements of the three sets $\{f \mid f : A \to B\}$, $\{f \mid f : A \to A\}$ and $\{f \mid f : B \to A\}$. State which functions are one-to-one and onto, and which are bijections.

Exercise 35 Determine which of the following relations are functions. For those which are functions, determine whether they are one-to-one and onto. Also, give the inverse function when it exists. Justify your answer throughout.

1) $\{\langle m,n\rangle \in \mathbb{N}^2 \mid m = n\}$;
2) $\{\langle m,n\rangle \in \mathbb{N}^2 \mid m \neq n\}$;
3) $\emptyset \subseteq \mathbb{N}^2$;
4) $\{\langle a,a\rangle,\langle a,b\rangle,\langle a,c\rangle\} \subseteq \{a,b,c\}^2$;
5) $\{\langle a,a\rangle,\langle b,a\rangle,\langle c,a\rangle\} \subseteq \{a,b,c\}^2$;
6) $\{\langle a,b\rangle,\langle b,c\rangle,\langle c,a\rangle\} \subseteq \{a,b,c\}^2$;
7) $\{\langle x,y\rangle \in \mathbb{Z}^2 \mid x = y \vee x = -y\}$;
8) $\{\langle x,y\rangle \in \mathbb{R}^2 \mid y = 1/(x^2-4)\}$;
9) $\{\langle x,y\rangle \in \mathbb{Z}^2 \mid x^2 = y\}$.

Exercise 36 1) Let $A = \{a,b,\{b\}\}$ and $B = \{\{a\},\{\emptyset\}\}$. How many functions are there from A to B? How many of these functions are onto? How many one-to-one? How many partial functions are there from A to B?

2) Now let A and B be arbitrary sets with $|A| = m$ and $|B| = n$. How many functions are there from A to B, and how many partial functions?

Exercise 37 Let $f : A \rightarrow B$ and $g : B \rightarrow C$ be functions.

1) Prove that if $g \circ f$ is one-to-one then so is f.
2) Give a specific example of f and g such that $g \circ f$ is one-to-one but g is not.
3) Prove that if $g \circ f$ is onto then so is g.
4) Give a specific example of f and g such that $g \circ f$ is onto but f is not.

Exercise 38 Assume that $A_1 \approx A_2$ and $B_1 \approx B_2$; show that $A_1 \times B_1 \approx A_2 \times B_2$.

* *Exercise 39 Let A, B and C be arbitrary sets. If possible, find explicit bijections between the following pairs of sets. Justify your answer. If this is not possible in general, explain why by analysing special cases.*

1) $A \times B$ *and* $B \times A$;
2) A^2 *and* A;
3) $((A \rightarrow B) \rightarrow C)$ *and* $(A \rightarrow (B \rightarrow C))$;
4) $((A \times B) \rightarrow C)$ *and* $(A \rightarrow (B \rightarrow C))$.

5 Computable functions

We now briefly look at *computable* functions. The need for a well-defined notion of computation arose in the early 20^{th} century, when mathematicians were looking to understand what it meant to be able to decide if a statement was true or false. This is known as Hilbert's *Entscheidungsproblem*.

The intuitive notion of computation and computability is relatively clear: computability is the ability to solve a problem in an effective manner in finitely many steps and any sequence of operations on objects that results in an answer can be considered a computation. For example, $\sqrt{2}$ is not computable, since it needs an infinite process to establish its value, whereas the square function $f(x) = x^2$ on $\mathbb{N}$ always returns an answer, and can easily be defined using Peano arithmetic.

Formal notions of computability were defined in various forms by David Hilbert, Kurt Gödel, Alonzo Church, Stephen Kleene, Emil Post, Alan Turing, and Andrey Markov, all in the first half of the last century. These (very) different notions were shown to all be equivalent (each can be shown to be *Turing complete*, which means that all computable functions can be encoded in them), which led to Church's thesis, which says that there is *only one* notion of computability.[16]

We start with presenting Gödel's first attempt to defining computable functions. His definition became later known as that of primitive recursive functions; these were known to Hilbert circa 1890 and were formally defined by Gödel in 1929.

Definition 5.1 (PRIMITIVE RECURSIVE FUNCTIONS) The class of *primitive recursive* functions $\mathcal{F}_{pr}$ in $\mathbb{N}^k \rightarrow \mathbb{N}$ (for any k) is generated by:

[16] This thesis gets occasionally challenged, but so far has not been shown incorrect.

$$\begin{array}{llcll}
(\textit{Initial functions}): & \textit{Zero}(x) & = & 0 & \textit{Zero Function} \\
& \textit{Succ}(x) & = & x+1 & \textit{Successor} \\
& \textit{Proj}^i_n(x_1,\ldots,x_n) & = & x_i \quad (1 \leq i \leq n) & \textit{Projection}
\end{array}$$

are in $\mathcal{F}_{pr}$.

$(\textit{Composition})$: If $g_1,\ldots,g_m : \mathbb{N}^k \to \mathbb{N}$, $h : \mathbb{N}^m \to \mathbb{N} \in \mathcal{F}_{pr}$, then $f : \mathbb{N}^k \to \mathbb{N}$ defined by

$$f(x_1,\ldots,x_k) = h(g_1(x_1,\ldots,x_k),\ldots,g_m(x_1,\ldots,x_k))$$

is in $\mathcal{F}_{pr}$.

$(\textit{Recursion})$: If $g : \mathbb{N}^k \to \mathbb{N}$ and $h : \mathbb{N}^{k+2} \to \mathbb{N} \in \mathcal{F}_{pr}$, then $f : \mathbb{N}^{k+1} \to \mathbb{N}$ defined by

$$\begin{array}{rcl}
f(0,x_1,\ldots,x_k) & = & g(x_1,\ldots,x_k) \\
f(\textit{Succ}(y),x_1,\ldots,x_k) & = & h(y,f(y,x_1,\ldots,x_k),x_1,\ldots,x_k)
\end{array}$$

is in $\mathcal{F}_{pr}$.

For example, the following functions are primitive recursive:

$$\begin{array}{lrcl}
(\textit{Predecessor}): & \textit{Pred}(0) & = & 0 \\
& \textit{Pred}(\textit{Succ}(n)) & = & \textit{Proj}^1_2(n,\textit{Pred}(n))
\end{array}$$

Remark that the scheme forces us to use the recursion rule, since we cannot express the predecessor function using just *Zero*, *Succ*, or projections, but ignore the obsolete recursive call by just returning the first argument using a projection.

$$\begin{array}{lrcl}
(\textit{Addition}): & \textit{Add}(0,x) & = & \textit{Proj}^1_1(x) \\
& \textit{Add}(\textit{Succ}(y),x) & = & \textit{Succ}(\textit{Proj}^2_3(y,\textit{Add}(y,x),x))
\end{array}$$

Notice that, in the recursive scheme, we have used $h = \textit{Succ} \circ \textit{Proj}^2_3$.

$(\textit{Multiplication})$:

$$\begin{array}{rcl}
\textit{Mult}(0,x) & = & \textit{Proj}^1_1(0) \\
\textit{Mult}(\textit{Succ}(y),x) & = & \textit{Add}(\textit{Proj}^1_3(y,\textit{Mult}(y,x),x),\textit{Proj}^2_3(y,\textit{Mult}(y,x),x))
\end{array}$$

$$\begin{array}{lrcl}
(\textit{Factorial}): & \textit{Fact}(0) & = & \textit{Succ}(0) \\
& \textit{Fact}(\textit{Succ}(y)) & = & \textit{Mult}(\textit{Succ}(y),\textit{Fact}(y))
\end{array}$$

so here $k = 0$, $g() = \textit{Succ}(0)$ (so we see constants as parameterless functions), and $h(y,z) = \textit{Mult}(\textit{Succ}(y),z)$.

Notice that all primitive recursive functions are total.

In 1928 Ackermann published an example of a total function $\textit{Ack} : \mathbb{N}^2 \to \mathbb{N}$ that is an intuitively computable function, but is not primitive recursive; it is defined by:

$$\begin{array}{rcl}
\textit{Ack}(0,y) & = & \textit{Succ}(y) \\
\textit{Ack}(\textit{Succ}(x),0) & = & \textit{Ack}(x,1) \\
\textit{Ack}(\textit{Succ}(x),\textit{Succ}(y)) & = & \textit{Ack}(x,\textit{Ack}(\textit{Succ}(x),y))
\end{array}$$

It is computable (in fact, in Example 7.12 we will show it always terminates) but does not belong to $\mathcal{F}_{pr}$.

In order to solve this problem, Kleene added the notion of *minimisation*, defining the class of *(partial) recursive functions* in 1934,.

Definition 5.2 ((Partial) Recursive Functions) The class of *(partial) recursive* functions $\mathcal{F}_{rec}$ in $\mathbb{N}^k \to \mathbb{N}$ (for any k) is defined as the set of primitive recursive functions, but adding:

4) If $f : \mathbb{N}^{n+1} \to \mathbb{N} \in \mathcal{F}_{rec}$, then $h : \mathbb{N}^n \to \mathbb{N} \in \mathcal{F}_{rec}$, where h is defined by:

$$h(x_1,\ldots,x_n) = \mu y\ (f(y,x_1,\ldots,x_n) = 0)$$

Intuitively, minimisation 'μy' expresses the search for the smallest argument y that causes $f(y,x_1,\ldots,x_n)$ to return 0, beginning with 0 and proceeding upwards, so stepping through

$$f(0,x_1,\ldots,x_n),\ f(1,x_1,\ldots,x_n),\ f(2,x_1,\ldots,x_n),\ f(3,x_1,\ldots,x_n),\ \textit{etc.}$$

If there no such argument exists, the search never terminates; hence h might be not defined on all inputs, and would then be partial.

It might be surprising, but this is all. By Church's thesis, any program in any programming language can be written as a a partial recursive function – not that you would want to do that, but it is possible.

6 On infinity

We will explore examples of infinite sets in two stages. We first look at those sets which have the same cardinality as the natural numbers, since such sets have nice properties. We then briefly explore examples of infinite sets which are truly bigger than the set of natural numbers.

The set of natural numbers is one of the simplest infinite sets. Using Peano's axioms we can build it up by stages:

$$\begin{array}{l} 0 \\ 0,1 \\ 0,1,2 \\ \vdots \end{array}$$

in such a way that any number n will appear at some stage. We distinguish those sets which are either finite or which have a bijection to $\mathbb{N}$.

6.1 Countable sets

We first introduce the notion of countability of a set A, which expresses that we can effectively list the elements of A, albeit perhaps in an infinite list: $A = \{a_0, a_1, a_2, a_3, \ldots\}$.

Definition 6.1 (COUNTABILITY) A set A is *countable* if A is finite or $A \approx \mathbb{N}$.

Another way to define this is to say that A is countable if there exists a surjective function from $\mathbb{N}$ to A. Then A is finite, or if there exists a surjective function from A to $\mathbb{N}$, then by Cantor-Bernstein's theorem (Theorem 4.14) we have $A \approx \mathbb{N}$.

The following properties hold:

Proposition 6.2 1) A set A is countable if and only if there exists a surjection $f : \mathbb{N} \to A$.
2) Let A be a countable set; if $B \subseteq A$, then B is countable.

Example 6.3 ($\mathbb{Z}$ IS COUNTABLE) The integers $\mathbb{Z}$ are countable, since they can be listed as:

$$0, -1, 1, -2, 2, -3, 3, \ldots .$$

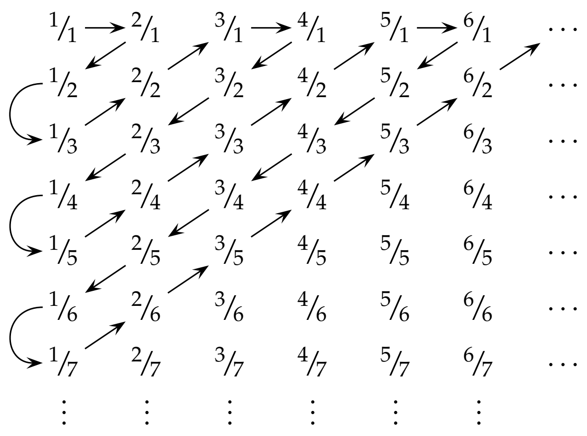
This counting function can be defined formally by the bijection $g : \mathbb{N} \to \mathbb{Z}$ defined by

$$g(x) = \begin{cases} x/2 & (x \textit{ even}) \\ -(x+1)/2 & (x \textit{ odd}) \end{cases}$$

Our bijection does not respect the order - but of course this is not necessary.

By Example 4.33 we know that $\mathbb{N}^2$ is countable.

Example 6.4 ($\mathbb{Q}$ IS COUNTABLE) Since we can represent (positive) rational numbers as fractions of natural numbers, by Example 4.33 the positive rational numbers are also countable. Following the approach above, we can easily define a function from $\mathbb{N}$ to $\mathbb{Q}^+$ (the set $\{q \in \mathbb{Q} \mid q > 0\}$) that is *onto*:



Notice that this diagram does not define a bijection itself, since $^1/_2 = {}^2/_4 = {}^3/_6$, etc; in fact, *every* fraction will appear *infinitely many times* in the image of f. For all this *infinite waste*, it is clear however that for every $r \in \mathbb{Q}^+$ there will be an n such that $f(n) = r$, making f *onto*. Since the function $h : \mathbb{Q}^+ \to \mathbb{N}$, defined by $h(^p/_q) = p - 1$ is onto as well, using Cantor-Bernstein we know that $\mathbb{Q}^+ \approx \mathbb{N}$, so $\mathbb{Q}^+$ is countable.

Now let $f : \mathbb{N} \to \mathbb{Q}^+$ be this counting. It is now straightforward to build a counting g of $\mathbb{Q}$ as follows:

$$g(n) \; = \; \begin{cases} 0 & (n = 0) \\ f(n/2) & (n \textit{ even}) \\ -f((n{-}1)/2) & (n \textit{ odd}) \end{cases}$$

So also $\mathbb{Q}$ is countable.

We will now show that the set of all *finite* subsets of $\mathbb{N}$ is countable.

Example 6.5 The set of finite subsets of $\mathbb{N}$, defined as $\{V \subseteq \mathbb{N} \mid \exists n \in \mathbb{N} \, (\, |V| = n)\}$, is countable. We will show this property first for the set of finite subsets of $\mathbb{N} \setminus \{0\}$,

$$\wp_f(\mathbb{N} \setminus \{0\}) \; = \; \{V \subseteq \mathbb{N} \setminus \{0\} \mid \exists n \in \mathbb{N} \, (\, |V| = n)\},$$

using Theorem 4.26.

Let $V \in \wp_f(\mathbb{N} \setminus \{0\})$ be a finite set and $|V| = n$, then we can write V as a list

$$V \; = \; v_0 \, v_1 \, v_2 \, v_3 \, \cdots \, v_n$$

such that $v_i < v_j$ for $i \leq j$. Let $\{p_1, p_2, p_3, \ldots\}$ be the set of prime numbers such that $p_i < p_j$ for $i < j$. We define $f : \wp_f \, \mathbb{N} \to \mathbb{N}$ by:

$$f(V) \; = \; 2^{v_1} \times 3^{v_2} \times 5^{v_3} \times 7^{v_4} \times \cdots \times p_n^{v_n} \; = \; \Sigma_{i=1}^{n} p_i^{v_i}$$

(notice that $f(\emptyset) = 1$). Since each number has its unique decomposition as a product of prime numbers, it is straightforward to verify that f is an injection. Note that $g : \mathbb{N} \to \wp_f(\mathbb{N} \setminus \{0\})$, defined through $g(n) = \{n+1\}$ is an injection as well, so by Theorem 4.26 we have $\mathbb{N} \approx \wp_f(\mathbb{N} \setminus \{0\})$. Mark that

$$\{V \subseteq \mathbb{N} \mid \exists n \in \mathbb{N} \, (\, |V| = n)\} \; \approx \; \{V \subseteq \mathbb{N} \setminus \{0\} \mid \exists n \in \mathbb{N} \, (\, |V| = n)\}$$

through the bijection $g(V) = \{m+1 \mid m \in V\}$.

6.2 Uncountable sets

Using his diagonalisation argument, Cantor in fact showed that there are *uncountable sets*: that is, infinite sets that are too large to be countable. In particular, his theorem implies that $\mathbb{N} \not\approx \wp \, \mathbb{N}$, so immediately we know that $\wp \, \mathbb{N}$ is *not* countable. To illustrate the diagonalisation argument, we look at Cantor's proof in detail for the case of natural numbers:

Example 6.6 ($\wp \, \mathbb{N}$ IS NOT COUNTABLE) We will show that any list of subsets of $\mathbb{N}$ will be missing a subset, so no mapping of $\mathbb{N}$ to $\wp \, \mathbb{N}$ can be onto, so no bijection between $\mathbb{N}$ and $\wp \, \mathbb{N}$ can exist, as predicted by Cantor's theorem.

Through its characteristic function (see Definition 4.6), we can write any subset $V \subseteq \mathbb{N}$ (or $V \in \wp \, \mathbb{N}$) as a list of 0s and 1s as

$$V \; = \; v_0 \, v_1 \, v_2 \, v_3 \, \cdots$$

where every $v_i = 1$ if $i \in V$, and $v_i = 0$ if $i \notin V$.

Assume now that there exists a bijection $f : \mathbb{N} \to \wp \, \mathbb{N}$; we write V_i for $f(i)$, and represent f by the list $V_0, V_1, V_2, V_3, \ldots$. We can now define a set W through its characteristic function

$$w_0\, w_1\, w_2\, w_3\, \cdots$$

by: $w_i = 1 - v^i_i$ (i.e. $w_i = 1$ if $v^i_i = 0$, and $w_i = 0$ if $v^i_i = 1$). We can represent this through the diagram

$$\begin{array}{lccccccccc}
V_0 = & v^0_0 & v^0_1 & v^0_2 & v^0_3 & v^0_4 & v^0_5 & v^0_6 & v^0_7 & \cdots \\
V_1 = & v^1_0 & v^1_1 & v^1_2 & v^1_3 & v^1_4 & v^1_5 & v^1_6 & v^1_7 & \cdots \\
V_2 = & v^2_0 & v^2_1 & v^2_2 & v^2_3 & v^2_4 & v^2_5 & v^2_6 & v^2_7 & \cdots \\
V_3 = & v^3_0 & v^3_1 & v^3_2 & v^3_3 & v^3_4 & v^3_5 & v^3_6 & v^3_7 & \cdots \\
V_4 = & v^4_0 & v^4_1 & v^4_2 & v^4_3 & v^4_4 & v^4_5 & v^4_6 & v^4_7 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
W = & 1-v^0_0 & 1-v^1_1 & 1-v^2_2 & 1-v^3_3 & 1-v^4_4 & 1-v^5_5 & 1-v^6_6 & 1-v^7_7 & \cdots
\end{array}$$

Then clearly $W \subseteq I\!N$; also, W differs from each set in the list 'on the diagonal':

$$i \in W \Leftrightarrow i \notin V_i = i \notin f(i)$$

exactly as demanded by Cantor's construction. So clearly $W \neq V_i$ for all $i \in I\!N$, since they differ on the ith element, and therefore W is not in our list. This is independent of the list we started with, so any list we give will be at least missing one element. So f is not a bijection, so no bijection between $I\!N$ and $\wp\, I\!N$ can exist.

So we cannot list all the subsets of $I\!N$, and $\wp\, I\!N$ is not countable; the sets are both infinitely large, but with different kinds of infinity.

Another important example of an uncountable set is the set of real numbers $I\!R$. Using the diagonalisation argument, to show that this set is also not countable, we will show that every list of real numbers from the interval $[0,2]$ is at least missing one element.

Example 6.7 ($I\!R$ IS NOT COUNTABLE) Remark that we can write any number $a \in [0,2] \subseteq I\!R$ (all real numbers in between and including 0 and 2) in binary notation as an infinite sequence

$$a = a_0 \times 2^0 + a_1 \times 2^{-1} + a_2 \times 2^{-2} + a_3 \times 2^{-3} + \cdots = \Sigma^{\infty}_{i=0} a_i 2^{-i}$$

where every $a_i \in \{0,1\}$, and represent a by $a_0a_1a_2a_3\cdots$. Notice that 2 is represented by an infinite sequence of 1s (since $2 = \Sigma^{\infty}_{i=0} 2^{-i} = 1/1 + 1/2 + 1/4 + 1/8 + 1/16 + \cdots$), and 0 by an infinite sequence of 0s. Moreover, there are numbers that have *more than one* representation: for example, $1.5 = 11000\cdots = 101111\cdots$; these are called the *dyadic rationals*, and are of the form $n/2^k$. These 'doubles' have all the characteristic that they end with an infinite number of 0s or 1s; we call this *a 0-tail* (or 1-tail); these create a complication in the diagonalisation argument we will use, but that we will deal with successfully.

Now assume $[0,2]$ is countable, and let a^0, a^1, a^2, a^3, ... be a list of all the elements of $[0,2]$. We can now use the diagonalisation technique and construct a number not in the list, but have to be a bit more careful, seen that we need to avoid to create an alternative representation for a dyadic rational. So we 'skew' the argument, and define the number b such that, for every $i \in I\!N$:

$$\begin{array}{lll}
b_{2i} & = 1 - a^i_{2i} & \\
b_{2i+1} & = a^i_{2i} & = 1 - b_{2i}
\end{array}$$

Note that, for all i, if $b_{2i} = 0$, then $b_{2i+1} = 1$, and if $b_{2i} = 1$, then $b_{2i+1} = 0$.

We can represent the list a^0, a^1, a^2, a^3, ... through the diagram

$$\begin{array}{lccccccccccc}
a^0 = & a^0_0 & a^0_1 & a^0_2 & a^0_3 & a^0_4 & a^0_5 & a^0_6 & a^0_7 & a^0_8 & a^0_9 & \cdots \\
a^1 = & a^1_0 & a^1_1 & a^1_2 & a^1_3 & a^1_4 & a^1_5 & a^1_6 & a^1_7 & a^1_8 & a^1_9 & \cdots \\
a^2 = & a^2_0 & a^2_1 & a^2_2 & a^2_3 & a^2_4 & a^2_5 & a^2_6 & a^2_7 & a^2_8 & a^2_9 & \cdots \\
a^3 = & a^3_0 & a^3_1 & a^3_2 & a^3_3 & a^3_4 & a^3_5 & a^3_6 & a^3_7 & a^3_8 & a^3_9 & \cdots \\
a^4 = & a^4_0 & a^4_1 & a^4_2 & a^4_3 & a^4_4 & a^4_5 & a^4_6 & a^4_7 & a^4_8 & a^4_9 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
b = & 1-a^0_0 & a^0_0 & 1-a^1_2 & a^1_2 & 1-a^2_4 & a^2_4 & 1-a^3_6 & a^3_6 & 1-a^4_8 & a^4_8 & \cdots
\end{array}$$

Then it will be clear that $b \in [0,2]$, and $b \neq a^i$, for all $i \in \mathbb{N}$, since they differ in position $2i$; so b itself as a sequence is not in the list. So far, so good.

Did we create an alternative representation of a dyadic rational? Can it be that b is *equal* to one of the elements of the list, say a_m, in the sense that they are different representations of the same number? As remarked above, this is possible only if b has a 0-tail or a 1-tail (so the skewed diagonal has a 1-tail or a 0-tail). But this is impossible, since, by construction, every pair of digits b_{2i} and b_{2i+1} in b contains both a 0 and a 1. So with b we have created a new number that does not represent a number already in our list.

This is independent of the list we started with, so any list we give will be at least missing one element, so we cannot list all the real numbers in $[0,2]$. So we cannot count all real numbers in the interval $[0,2]$, and therefore certainly we cannot count $\mathbb{R}$.

We can also show that $\wp\,\mathbb{N} \approx \mathbb{R}$. Since we know that $\mathbb{N} \not\approx \wp\,\mathbb{N}$, then also $\mathbb{N} \not\approx \mathbb{R}$.

Example 6.8 As in the previous example, we can represent any number $a \in [0,2] \subseteq \mathbb{R}$ via an infinite sequence $a_0a_1a_2a_3\cdots$ since

$$a \;=\; a_0 \times 2^0 + a_1 \times 2^{-1} + a_2 \times 2^{-2} + a_3 \times 2^{-3} + \cdots \;=\; \Sigma_{i=0}^{\infty} a_i 2^{-i}.$$

Define $f : [0,2] \to \wp\,\mathbb{N}$ by:

$$f(a_0a_1a_2a_3\cdots) \;=\; \{i \in \mathbb{N} \mid a_i = 1\}$$

and $g : \wp\,\mathbb{N} \to [0,2]$ by:

$$\begin{aligned} g(V) \;&=\; v_0v_1v_2v_3\cdots \\ &\qquad \textit{where } v_i = \chi_V(i). \end{aligned}$$

It is easy to show that f and g are each other's inverse:

$$\begin{aligned} g \circ f(a_0a_1a_2a_3\cdots) &= g(\{i \in \mathbb{N} \mid a_i = 1\}) \\ &= v_0v_1v_2v_3\cdots && (v_i = \chi_V(i) \wedge V = \{i \in \mathbb{N} \mid a_i = 1\}) \\ &= v_0v_1v_2v_3\cdots && (v_i = 1 \textit{ if } a_i = 1 \textit{ and } v_i = 0 \textit{ if } a_i = 0) \\ &= a_0a_1a_2a_3\cdots \end{aligned}$$

and

$$\begin{aligned} f \circ g(V) &= f(v_0v_1v_2v_3\cdots) && (v_i = \chi_V(i)) \\ &= \{i \in \mathbb{N} \mid v_i = 1\} && (v_i = \chi_V(i)) \\ &= \{i \in \mathbb{N} \mid \chi_V(i) = 1\} \\ &= V. \end{aligned}$$

So $[0,2] \approx \wp\,\mathbb{N}$; using $h(x) \;=\; \tan(\pi \times (^1\!/_2 x - {}^1\!/_2))$ we have a surjection from $[0,2]$ onto $\mathbb{R}$, and with

$$k(x) \;=\; \begin{cases} x & (x \in [0,2]) \\ 0 & (\textit{otherwise}) \end{cases}$$

we have a surjection from $\mathbb{R}$ to $[0,2]$, so $[0,2] \approx \mathbb{R}$; so also $\mathbb{R} \approx \wp\,\mathbb{N}$.

The difference between integers and reals gives a complication for computer science: we cannot manipulate reals in the way we can natural numbers. Instead, we have to use approximations, such as the floating point decimals (given by the type `Float` in Haskell).

6.3 Different infinities

So we know that $\mathbb{R}$ is really larger than $\mathbb{N}$ and $\mathbb{Q}$, but the proof suggest this is only 'by a bit'. Nothing could be farther from the truth. Although in-between each pair of rational numbers there exists a real number, and in between each pair of real numbers there exists a rational number, the measure of the sets $\mathbb{Q}$ and $\mathbb{R}$ is very different.

To illustrate the difference between $|\mathbb{Q}| = |\mathbb{N}|$ and $|\mathbb{R}|$, we can show that $\mathbb{Q}$ is 'sparse' in $\mathbb{R}$. We will achieve this by defining a subset $V \subseteq \mathbb{R}$ whose 'size' is arbitrarily small and contains all rationals, so $\mathbb{Q} \subseteq V$ and $\mathbb{R} \setminus V$ is a set containing only pure reals. In other words, given any positive size, no matter how small, we will be able to create a set with that size that contains all rationals.

Example 6.9 ($\mathbb{Q}$ IS A NULL SET IN $\mathbb{R}$) We know from Example 6.4 that we can count the rationals, *i.e.* put them in an exhaustive list; so let q_0, q_1, q_2, ... be such a list. Now we cast an interval in $\mathbb{R}$ around each element of $\mathbb{Q}$, and define:

$$\begin{array}{rcl} V_\delta^i & \triangleq & (q_i - \delta \times 2^{-i},\, q_i + \delta \times 2^{-i}) \\ V_\delta & \triangleq & \cup_{i=0}^{\infty} V_\delta^i \end{array}$$

Remark that each V_i is an interval of size $2 \times \delta \times 2^{-i} = 2^{-i+1}\delta$; we write $\| V_\delta^i \|$ for this size; notice that $\mathbb{Q} \subseteq V_\delta$.

Now the total size of V_δ, $\| V_\delta \|$, is strictly greater than 0 and smaller or equal to the sum of all the sizes of the V_δ^is (they might overlap, of course), i.e.

$$0 \;<\; \| V_\delta \| \;\leq\; \Sigma_{i=0}^{\infty} \| V_\delta^i \| \;=\; \Sigma_{i=0}^{\infty} 2^{-i+1}\delta \;=\; 4\delta$$

Notice that this is true for any δ. Since we can make δ as small as we like, this essentially shows that $V = \lim_{\delta \to 0} V_\delta$ is negligible, but still non-empty (also called a *null* or *zero* set). But since $\mathbb{Q} \subseteq V$, so is $\mathbb{Q}$.

So, for any size, however small, we can put $\mathbb{Q}$ in a subset of $\mathbb{R}$ of that size; the rest of the real line contains only true reals. So basically, within the space of $\mathbb{R}$, $\mathbb{Q}$ occupies no space at all; the rationals do not matter in comparison: the difference in levels of infinity between $\mathbb{Q}$ (or $\mathbb{N}$) and $\mathbb{R}$ is enormous. Moreover, the set of *irrational* numbers, defined as $\mathbb{R} \setminus \mathbb{Q}$, is equivalent to $\mathbb{R}$: $\mathbb{R} \setminus \mathbb{Q} \approx \mathbb{R}$.

Through Example 6.6 and 6.7 we have essentially shown that we can define a bijection between the interval $[0,2]$ and $\wp\,\mathbb{N}$ (taking care of the dyadic rationals), so also the difference in levels of infinity between $\mathbb{N}$ and $\wp\,\mathbb{N}$ is enormous.

So there are two fundamentally different kinds of infinity here: the cardinality of $\mathbb{N}$, $|\mathbb{N}|$ called $\aleph_0$, and that of $\wp\,\mathbb{N}$, called $\aleph_1$. Notice that by Cantor's theorem we have an infinite hierarchy of notions of infinity:

$$\mathbb{N} \;\not\approx\; \wp\,\mathbb{N} \;\not\approx\; \wp(\wp\,\mathbb{N}) \;\not\approx\; \wp(\wp(\wp\,\mathbb{N})) \;\not\approx\; \wp(\wp(\wp(\wp\,\mathbb{N}))) \;\cdots\;.$$

and

$$|\mathbb{N}| \;<\; |\wp\,\mathbb{N}| \;<\; |\wp(\wp\,\mathbb{N})| \;<\; |\wp(\wp(\wp\,\mathbb{N}))| \;<\; |\wp(\wp(\wp(\wp\,\mathbb{N})))| \;\cdots\;.$$

Since $\wp\,\mathbb{N} \approx \mathbb{R}$, also $\wp(\wp\,\mathbb{N}) \approx \wp\,\mathbb{R}$, etc.

The *continuum hypothesis* states that there is no different notion of infinity in between $\aleph_0$ and $\aleph_1$, an hypothesis that can neither be proved nor disproved within present set theory.

Exercises

Exercise 40 *1) Assume that $A \approx B$; show that $A^2 \approx B^2$.*

2) Assume that the set A is countable; show that A^2 is as well.

Exercise 41 *1) Prove that $\{0,1\} \times \mathbb{N}$ is countable.*

2) Use this property to prove that if the disjoint sets X and Y are countable, then so is $X \cup Y$.

3) Now prove that if X and Y are countable then so is $X \cup Y$.

Exercise 42 *Show that the sets $\mathbb{Z}^2$ and $\mathbb{N}^3$ are countable.*

Exercise 43 *Show that a countable union of countable sets is countable.*

Exercise 44 *Show $\{0,1\}^V \approx \wp(V)$.*

Exercise 45 *Show that $\{0,1\}^{\mathbb{N}}$ is not countable.*

Exercise 46 *One of the following sets is finite, one countably infinite and one uncountable. Determine which is which, and justify your answer.*

1) $\{f : \mathbb{N} \to \{0,1\} \mid \forall n \in \mathbb{N}\; (f(n) \leq f(n+1))\}$

2) $\{f : \mathbb{N} \to \{0,1\} \mid \forall n \in \mathbb{N}\; (f(n) \neq f(n+1))\}$

3) $\{f : \mathbb{N} \to \mathbb{N} \mid \forall n \in \mathbb{N}\; (f(n) \leq f(n+1))\}$

7 Orderings

Orderings are special relations which characterise when one object is greater than (in any sense of the word) another. Orderings on sets of numbers such as $\mathbb{N}$, $\mathbb{Z}$ and $\mathbb{R}$ are familiar: the ordering '$<$' describes the *less than* ordering, and '$\leq$' the *less than or equal* ordering. We can define orderings on all sets; here we give the formal definitions and properties of some orderings.

Definition 7.1 (ORDERINGS) Let R be a binary relation on a set A. We distinguish the following different notions of orders.

(R *is a* pre-order): R is reflexive and transitive (so not necessarily symmetric).

(R *is* anti-symmetric): $\forall a,b \in A\ (x\,R\,y \wedge y\,R\,x \Rightarrow x =_A y)$.

(R *is a* partial order *relation*): R is reflexive, transitive, and anti-symmetric. Partial orders are often denoted by $\leq$ and we write $(A,\leq)$ or $\leq_A$ to denote a partial order $\leq$ on A.

(R *is* irreflexive): $\forall a \in A\ (\neg(a\,R\,a))$.

(R *is a* strict partial order *relation*): R is irreflexive and transitive. Strict partial orders are often denoted by $<$.

(R *is a* total order): A partial order that also satisfies: $\forall a,b \in A\ (a\,R\,b \vee b\,R\,a)$. Total orders are also sometimes called *linear* orders.

Remember that if R is symmetric then, for all x and y, if $\langle x,y\rangle \in R$, then also $\langle y,x\rangle \in R$. Then anti-symmetry expresses that a relation is nowhere symmetric other than in a trivial way: for no pair $\langle x,y\rangle$ in R with $x \neq y$ is $\langle y,x\rangle$ also in R. Notice, however, that anti-symmetric is not the opposite of symmetric, since a relation can be both symmetric and anti-symmetric: take for example the identity relation or the empty relation.

Similarly, irreflexive is not the same as not reflexive; irreflexive means that the relation is non-reflexive on all pairs ($\forall a \in A\ (\neg(a\,R\,a))$), and not reflexive means that there is at least one non-reflexive pair ($\neg\forall a \in A\ (a\,R\,a) = \exists a \in A\ (\neg(a\,R\,a))$); notice the difference in quantifier.

Example 7.2 • Using Definition 3.1, the (numerical) order $\leq_{\mathbb{N}}$ is defined as the reflexive, transitive closure of the relation $\{\langle n,m\rangle \in \mathbb{N}^2 \mid m = \mathit{Succ}(n)\}$; then $\leq_{\mathbb{N}}$ is a total order on $\mathbb{N}$.

• Using Definition 3.4, we can now define $\leq_{\mathbb{Z}}$ by:

$$\leq_{\mathbb{Z}} \ \triangleq \ \begin{cases} \{\langle n,m\rangle \mid n \leq_{\mathbb{N}} m\} \cup \\ \{\langle -m,-n\rangle \mid n \leq_{\mathbb{N}} m\} \cup \\ \{\langle -n,m\rangle \mid n,m \in \mathbb{N}\} \end{cases}$$

and likewise

$$\leq_{\mathbb{Q}} \ \triangleq \ \{\langle\langle z_1,n_1\rangle,\langle z_2,n_2\rangle\rangle \in (\mathbb{Z}\times\mathbb{N})^2 \mid z_1 \times n_2 \leq_{\mathbb{Z}} z_2 \times n_2\}$$

• Formally defining $\leq_{\mathbb{R}}$ is a bit more work, and out of scope for this course.

Example 7.3 *1*) Division on $\mathbb{N} \setminus \{0\}$ induces a partial order: define

$$\leq_{/} \ \triangleq \ \{\langle n,m\rangle \in \mathbb{N}^2 \mid \exists k\ (m = k \times n)\}$$

then $\leq_{/}$ is a partial order.

2) For any set A, the power set of A ordered by subset inclusion, $(\wp A, \subseteq)$, is a partial order.

3) Suppose $(A,\leq_A)$ is a partial order and $B \subseteq A$. Then $(B,\leq_B)$ is a partial order, where $\leq_B$ denotes the restriction of $\leq_A$ to the set B^2.

$$\leq_B \ \triangleq \ \{\langle b_1,b_2\rangle \in B^2 \mid b_1 \leq_A b_2\}$$

4) Define a relation on formulae by: $A \leq B$ when $\vdash A \rightarrow B$. Then $\leq$ is a pre-order. For example, $\mathsf{false} \leq A \leq \mathsf{true}$ and $A \leq A \vee B$.

Definition 7.4 For any two partially ordered sets $(A,\leq_A)$ and $(B,\leq_B)$, there are two important orders on the product set $A \times B$:

$$\begin{array}{lrcl} \textit{Product order:} & \langle a_1,b_1\rangle \leq_P \langle a_2,b_2\rangle & \triangleq & (a_1 \leq_A a_2) \wedge (b_1 \leq_B b_2) \\ \textit{Lexicographic order:} & \langle a_1,b_1\rangle \leq_L \langle a_2,b_2\rangle & \triangleq & (a_1 <_A a_2) \vee (a_1 =_A a_2 \wedge b_1 \leq_B b_2) \end{array}$$

If $(A,\leq)$ and $(B,\leq)$ are both total orders, then the lexicographic order on $A\times B$ will be total. By contrast, the product order will in general only be partial.

For any partially ordered sets $(A,\leq)$ and $(B,\leq)$, the product order is contained in the lexicographic order.

Example 7.5 For any totally ordered (finite) set A, let A^* stand for the set $\{\epsilon\}\cup A\cup A^2\cup A^3\cup\cdots$, the set of all finite sequences made from elements of A, with ϵ denoting the empty sequence.

The *full lexicographic order* $\leq_F$ on A^* is defined as follows: given two sequences $u,v\in A^*$, if $u=\epsilon$ then $u\leq_F v$ and if $v=\epsilon$ then $v\leq_F u$. Otherwise, both u and v are non-empty so we can write $u=xu'$ and $v=yv'$ where x and y are the first symbols of u and v, respectively. So

$$u\leq_F v \;\triangleq\; (u=\epsilon)\vee(u=xu'\wedge v=yv'\wedge(x<_A y\vee(x=y\wedge u'\leq_F v')))$$

Remark that the lexicographic order gives the ordering for words in a dictionary, with the total order on characters that is defined by the alphabet.

7.1 Hasse Diagrams

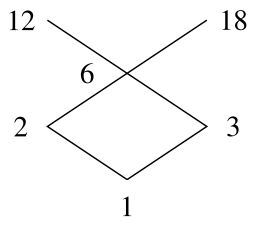
Since partial orders are special binary relations on a set, we can represent them by directed graphs. However, these graphs get rather cluttered if every arrow is drawn. We therefore introduce Hasse diagrams, which provide a compact way of representing the partial order. First we require some definitions.

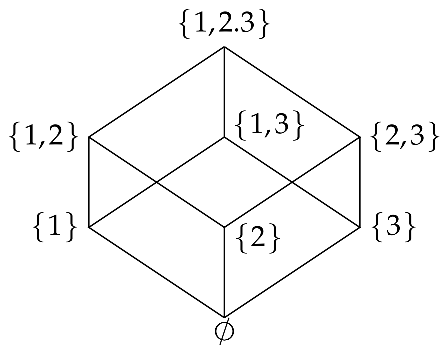
Definition 7.6 If R is a partial order on a set A and $a\,R\,b$ for $a\neq b$, we call a a *predecessor of* b, and similarly b a *successor of* a.

If a is a predecessor of b and there exists no $c\neq a,b$ with $a\,R\,c$ and $c\,R\,b$, then a is the *immediate predecessor* of b.

Hasse diagrams are like directed graphs, except that they just record the immediate predecessors; the other pairs in the partial order can be inferred. Also, the direction of the lines is usually omitted, with the convention that all lines are directed 'up the page'. We give two examples of Hasse diagrams.

Example 7.7 For example, the Hasse diagram for the relation 'is a divisor of' for the set $\{1,2,3,6,12,18\}$ and for the binary relation $\subseteq$ on $\wp\{1,2,3\}$ are, respectively





7.2 Analysing Partial Orders

We will now look at properties of partial orders.

Definition 7.8 Let (A,R) be a partial order and $a\in A$.

$$\begin{aligned} a \text{ is } \textit{minimal} &\;\triangleq\; \forall b\in A\;(b\,R\,a\Rightarrow b=_A a)\\ a \text{ is } \textit{least} &\;\triangleq\; \forall b\in A\;(a\,R\,b)\\ a \text{ is } \textit{maximal} &\;\triangleq\; \forall b\in A\;(a\,R\,b\Rightarrow a=_A b)\\ a \text{ is } \textit{greatest} &\;\triangleq\; \forall b\in A\;(b\,R\,a)\end{aligned}$$

Remember that, since R is a partial order, it is reflexive, so $\forall a\in A\;(a\,R\,a)$.

In Example 7.7, the least (and minimal) element of the first relation is 1, the maximal elements are 12 and 18, and there exists no greatest element. The least (and minimal) element of the second relation is $\emptyset$, and the greatest (and maximal) element is $\{1,2,3\}$.

With the usual partial order $(I\!N,\leq)$, the least element is 0, and there exists no maximal element. The partial orders $(I\!N,\leq)$ and $(\mathbb{Z},\leq)$ are different from each other: the latter has no least element.

There can be many minimal elements: take $W = \{V \subseteq I\!N \mid |V| \geq 1\}$, then each $\{n\}$ is minimal in $(W,\subseteq)$ and there exists no least element.

Proposition 7.9 Let $(A,\leq)$ be a partial order.

1) If $a \in A$ is a least element, then it is a minimal element.

2) If $a \in A$ is a least element, then it is unique.

3) If A is finite and non-empty, then $(A,\leq)$ must have a minimal element.

4) If $(A,\leq)$ is a total order, where A is finite and non-empty, then it has a least element.

Proof: To prove part (*1*), suppose that $a \in A$ is least and assume towards a contradiction that $b < a$ for some $b \in A$ such that $b \neq a$. But $a \leq b$ by definition of a being least. This contradicts anti-symmetry.

To prove part (*2*), suppose that a and b are both least elements. By definition of least element, we have $a \leq b$ and $b \leq a$. By anti-symmetry, it follows that $a = b$.

To prove part (*3*), pick any $a_1 \in A$. If a_1 is not minimal we can pick $a_2 < a_1$. If a_2 is not minimal, we can pick $a_3 < a_2$. In this way, we either find a minimal element, or get an infinitely long decreasing chain $a_1 > a_2 > a_3 > \cdots$. By construction, all the elements of the chain must be different, which contradicts the fact that A is a finite set.

Part (*4*) is left as Exercise 48. ■

7.3 Well-founded Partial Orders

Suppose we wish to prove that the Ackermann function

$$Ack(n,m) = \begin{cases} m+1 & (n=0) \\ Ack(n-1,\ 1) & (n>0 \wedge m=0) \\ Ack(n-1,\ Ack(n,m-1)) & (n>0 \wedge m>0) \end{cases}$$

always terminates, and therefore defines a total function. Counting down from n is not good enough, since the third equation does not decrease the first argument, because of the embedded $Ack(n,m-1)$. We will devise a different way of counting down, by defining a well-founded partial order with the property that it always decreases to a terminating state.

Definition 7.10 (WELL-FOUNDED PARTIAL ORDERS) A partial order $(A,\leq)$ is *well founded* if it has no infinite decreasing chain of elements: that is, for every infinite sequence $a_1, a_2, a_3, \ldots$ of elements in A with $a_1 \geq a_2 \geq a_3 \geq \cdots$, there exists $m \in I\!N$ such that $m \geq 1$ and $a_n = a_m$ for every $n \geq m$.

For example, the conventional numerical order $\leq_{I\!N}$ is a well-founded partial order. This is not the case for $\leq_{\mathbb{Z}}$, which can decrease for ever.

Proposition 7.11 If two partial orders $(A,\leq_A)$ and $(B,\leq_B)$ are well founded, then the lexicographical order $\leq_L$ on $A \times B$ (see Example 7.3) is also well founded.

Proof: Suppose $\langle a_1,b_1\rangle \geq_L \langle a_2,b_2\rangle \geq_L \langle a_3,b_3\rangle \geq_L \cdots$. Then $a_1 \geq_A a_2 \geq_A a_3 \geq_A \cdots$ by the definition of lexicographic ordering, so the sequence must ultimately consist of the same element, since $(A,\leq_A)$ is well founded. Therefore, there exists $m \in I\!N$ such that $a_n = a_m$ for every $n \geq m$.

Now by the definition of lexicographic order, we have $b_m \geq_B b_{m+1} \geq_B b_{m+3} \geq_B \cdots$, and this sequence must also ultimately end up being constant because $(B,\leq_B)$ is well founded. Thus, the original sequence is ultimately constant. ■

This result implies that the product order on $A \times B$ is well founded, since the product order is contained in the lexicographical order (see Example 7.3).

Example 7.12 Take the *Ack* function, and consider the strict lexicographical order on $I\!N^2$ by

$$\langle x,y\rangle < \langle x',y'\rangle \ \triangleq\ x < x' \vee (x = x' \wedge y < y')$$

Notice that, for any x and y,

$$\begin{array}{lll}\langle x+1,0\rangle & > \langle x,1\rangle & \\ \langle x+1,y+1\rangle & > \langle x,m\rangle & (\textit{for all } m) \\ \langle x+1,y+1\rangle & > \langle x+1,y\rangle & \end{array}$$

(notice that the second case implies $\langle x+1,y+1\rangle > \langle x, Ack(x+1,y)\rangle$) and so evaluating the *Ack* function takes us down the order. Moreover, the order is well founded, using Proposition 7.11 and the fact that $(\mathbb{N},<)$ is well founded. Even though an element such as $\langle 4,3\rangle$ has infinitely many other elements below it (for example, $\langle 3,y\rangle$ for every y), any decreasing chain must be finite. Hence, *Ack* is a total function and always returns an answer.

One of the more baffling things in mathematics (and a direct result of the axiom of choice, or, equivalently, directly implied by the axiom of well-ordering (see the next section)) is that there exists a well-ordering on $\mathbb{R}$; this is, of course, not the familiar strict partial order $<$. From the axiom of choice we can prove that it exists, but only by showing that if it does not exist, we get a contradiction. So we know that it cannot not exist, but do not know how it is defined; the proof is not *constructive* in that it does not provide the relation. Likewise, the well-ordering axiom just state the existence, not how to define it.

Exercises

Exercise 47 *1) Let $\leq_D$ denote a binary relation on the positive natural numbers, defined by $n \leq_D m \triangleq$ n divides m. Prove that $\leq_D$ is a partial order.*

2) Take $S = \{2,4,5,10,12,20,25\}$. Draw the Hasse diagram of the partial order $(S,\leq_D)$. Which elements are maximal, and which minimal?

3) Explain why the partial order in part (2) is not total. Find a total order $\leq_T$ on S such that $n \leq_D m$ implies $n \leq_T m$.

Exercise 48 *1) Let $(A,\leq)$ be a partial order; show that the relation $< \triangleq \{\langle a,b\rangle \mid a \leq b \wedge a \neq b\}$ is a strict partial order.*

2) Show that, if $(A,\leq)$ is a total order, where A is finite and non-empty, then it has a least element.

Exercise 49 *Let $(\wp(A),\subseteq)$ denote the standard partial order on the subsets of A, given by set inclusion. Take $S = \{a,b,c,d\}$.*

1) Determine whether there exists a greatest and least element of $(\wp A,\subseteq)$.

2) Draw the Hasse diagram of $(\wp S,\subseteq)$.

3) Explain why $(\wp S,\subseteq)$ is not total. Find a total order $(\wp S,\subseteq_T)$ such that $A \subseteq B$ implies $A \subseteq_T B$ for every $A,B \in \wp S$.

Exercise 50 *Take $A = \{1,2,3,4\}$, and let $\leq_L$ denote the lexicographic order on A^2 that is the natural extension of $\leq$ on A.*

1) Find all pairs in A^2 which are less than $\langle 2,3\rangle$.

2) Find all pairs in A^2 which are greater than $\langle 3,1\rangle$.

3) Draw the Hasse diagram of the order $(A^2,\leq_L)$.

* *Exercise 51* *Define orderings on the (total) functions in $\mathbb{N} \to \mathbb{N}$ as follows*

$$\begin{array}{lll} f \leq_1 g & \triangleq & \forall n\,(f(n) \leq g(n)) \\ f \leq_2 g & \triangleq & \exists m \forall n\,(n \geq m \Rightarrow f(n) \leq g(n)) \end{array}$$

1) Show that $\leq_1$ is a partial ordering. What is its least element?

2) Is $<_1$ well-founded? Justify your answer, by either producing an infinite descending chain or explaining why one cannot exist.

3) Show that $\leq_2$ is a pre-order. Explain why it is not a partial order.

* 8 Axioms for Set Theory

We shall now, for completeness sake, state the axioms for set theory as formulated by Zermelo and Fränkel. For reasons of legibility, we state these in English rather than as mathematical formulae.

Definition 8.1 (ZERMELO-FRÄNKEL AXIOMS FOR SET THEORY)

($\mathit{Axiom\ of\ Extensionality}$): Two sets are equal (are the same set) if they have the same elements: for all sets A, B, if for all z we have $z \in A$ when $z \in B$, then $A = B$.

($\mathit{Axiom\ of\ Regularity}$): (also called the *Axiom of Foundation*) Every non-empty set A contains a member y such that A and y are disjoint sets. (This implies, for example, that no set contains only itself.)

($\mathit{Axiom\ of\ Specification}$): (also called the *axiom schema of separation* or of *restricted comprehension*) Every subclass of a set that is defined through a predicate is itself a set: so if A is a set, and ϕ a formula in the language of set theory, then $\{x \in A \mid \phi(x)\}$ is a set as well.

($\mathit{Axiom\ of\ Union}$): (also called the axiom of pairing). For every set $\mathcal{A}$ of sets, the collection of all members from the sets in $\mathcal{A}$, $\{x \in B \mid B \in \mathcal{A}\}$ is a set.

($\mathit{Axiom\ of\ Replacement}$): The image of the domain set A under the definable function f (i.e. the range of f) falls inside a set B.

($\mathit{Axiom\ of\ Infinity}$): There exists a set that has infinitely many members.

($\mathit{Axiom\ of\ Power\ set}$): For any set A, there exists a set that contains every subset of A.

($\mathit{Axiom\ of\ Well\text{-}ordering}$): For any set A, there exists a binary relation R which is a well-founded total order on A.

9 Induction

Many of the definitions and properties we are to study in computing depend heavily on induction. Not only is the majority of our definitions inductive in nature, also most of the proofs are inductive. Since the kind of induction we use is of a more general nature than just induction over numbers, we will have a close look at the various notion of induction.

There are many different approaches to definitions of induction in the literature, not all clear or consistent. We try our best here.

9.1 Mathematical Induction

Using Peano's axiom of induction (Definition 3.1) we can show the following lemma:

Lemma 9.1 (MATHEMATICAL INDUCTION) *Let P be a unary predicate such that*

- $P(0)$ *is true, and*
- *for every natural number k, if $P(k)$ is true, then $P(\mathit{Succ}(k))$ is true.*

Then $P(n)$ is true for every natural number $n \in \mathbb{N}$:

Proof: Let $V \triangleq \{n \in \mathbb{N} \mid P(n)\}$. We will show that $0 \in V$ and that if $k \in V$, then $\mathit{Succ}(k) \in V$, for any $k \in \mathbb{N}$:

- Note that $P(0)$ holds by definition of P. So $0 \in V$.
- We need to show that $\mathit{Succ}(k) \in V$, so that $P(\mathit{Succ}(k))$ holds. Using the proof technique of induction we can assume that $k \in V$, so can assume that $P(k)$ holds. But if we can assume that $P(k)$ holds, then by definition of P, then also $P(\mathit{Succ}(k))$ holds, so $\mathit{Succ}(k) \in V$.

Then, by the principle of induction $\mathbb{N} \subseteq V$, so $\forall n \in \mathbb{N}\ (n \in V)$, so $\forall n \in \mathbb{N}\ (P(n))$. ■

The assumption $k \in V$ (used in the second part) is often called the *inductive hypothesis*. A common way to describe the second step in the above proof is then to say that it follows '*from the inductive hypothesis*' (abbreviated by *IH*), or '*by induction*'. Notice that it the assumption *is needed* to show that $Succ(k) \in V$; without it, there would be no way to show the result.

For example, we can show:

Lemma 9.2 Show that, for all $n \in \mathbb{N}$, $Add(n, Succ(0)) = Succ(n)$.

We leave this for Exercise 33.

Using our result from this lemma we can now state the principle of mathematical induction in Logic:

$$P(0) \wedge \forall k \in \mathbb{N}\,(P(k) \Rightarrow P(k+1)) \rightarrow \forall n \in \mathbb{N}\,(P(n))$$

So, to prove a property $P(x)$ for all natural numbers if suffices to show:

$(Base\ case)$: Prove $P(0)$.

$(Inductive\ Case)$: For every k, using the assumption that $P(k)$ holds, prove $P(k+1)$; in other words: prove $P(k) \rightarrow P(k+1)$.

These two proofs give you the 'right' to say that $P(n)$ holds for all n.

For example, the proof of Lemma 9.2 can then be stated as:

Proof: Let $P(n) \triangleq Add(n, Succ(0)) = Succ(n)$. We will show that $P(0)$ and that if $P(k)$, then $P(Succ(k))$, for all $k \in \mathbb{N}$.

- Note that $Add(0, Succ(0)) \triangleq Succ(0)$ by definition of *Add*. So $P(0)$.
- Assume that $P(k)$, so that $Add(k, Succ(0)) = Succ(k)$. We want to show that then $P(Succ(k))$, so that $Add(Succ(k), Succ(0)) = Succ(Succ(k))$. Well,

$$\begin{array}{rcl} Add(Succ(k), Succ(0)) & \triangleq & (by\ definition\ of\ Add) \\ Succ(Add(k, Succ(0))) & = & (IH) \\ Succ(Succ(k)) & & \end{array}$$

By Lemma 9.1, we have shown $\forall n \in \mathbb{N}\,(P(n))$, so $\forall n \in \mathbb{N}\,(Add(n, Succ(0)) = Succ(n))$. ■

Notice that not every (correct) statement over numbers is proved by induction:

Theorem 9.3 *There are infinitely many prime numbers.*

Proof: Take $\mathcal{P}$ to be the set of prime numbers. We will show that $|\mathcal{P}|$ is not finite, by showing that there is no n such that $|\mathcal{P}| = n$.

So, suppose $|\mathcal{P}| = n$, and let $\mathcal{P} = \{p_1, \ldots, p_n\}$. Define $m = (p_1 \times \cdots \times p_n) + 1$. Then either m is prime, and obviously $m \notin \mathcal{P}$, or there is a prime number p' and number q such that $m = p' \times q$. Now the question is: is $p' \in \mathcal{P}$? Suppose it is, so $p' = p_i$, for some $1 \leq i \leq n$. Since $p'_i \mid m$, also $p_i \mid (p_1 \times \cdots \times p_n) + 1$. Obviously $p_i \mid p_1 \times \cdots \times p_n$, so also $p_i \mid 1$. This is impossible, so $p' \notin \mathcal{P}$.

So there is no $n \in \mathbb{N}$ such that $|\mathcal{P}| = n$, so $|\mathcal{P}|$ is infinite.

9.2 Complete Induction

An alternative to the rule for induction is the principle of Complete Induction (also called *course of values* or *strong* induction):

Definition 9.4 (COMPLETE INDUCTION) To prove a property $P(n)$ for all natural numbers n, it suffices to show:

$(Base\ case)$: Prove $P(0)$.

$(Inductive\ Case)$: For every k, if $P(i)$ holds for every i smaller than or equal to k, then $P(k+1)$; in other words: $(\forall i \in \mathbb{N}\,(i \leq k \wedge P(i))) \rightarrow P(k+1)$.

Then $P(n)$ holds for all $n \in \mathbb{N}$.

This kind of induction is useful when $P(7)$ does not follow from $P(6)$, but from $P(2)$ and $P(5)$, say.

Theorem 9.5 *These two principles of induction coincide, i.e. accepting one principle you can show the other holds, and vice versa.*

Proof: The proof has two parts. First we show that Complete Induction implies Mathematical Induction, and then show the reverse.

- Let $P(n)$ be a property over n, and assume

$$P(0) \tag{1}$$
$$\forall k \in \mathbb{N}\ (P(k) \to P(k{+}1)) \tag{2}$$

  We have to show that $\forall n\ (P(n))$. To be able to reach this result using Complete Induction, we need to show first that

$$\forall k \in \mathbb{N}\ (\forall i \in \mathbb{N}\ (i \le k \wedge P(i)) \to P(k{+}1)) \tag{3}$$

  For every k, this is an implication, that is shown as follows: assume

$$\forall i \in \mathbb{N}\ (i \le k \wedge P(i)),$$

  then, in particular, $P(k)$, and, by assumption (2), we have $P(k{+}1)$. So

$$(\forall i \in \mathbb{N}\ (i \le k \wedge P(i))) \to P(k{+}1),$$

  for every k, so we have

$$\forall k \in \mathbb{N}\ ((\forall i \in \mathbb{N}\ (i \le K \wedge P(i))) \to P(k{+}1)).$$

  Since also $P(0)$ by (1), by course of values induction we get $\forall n\ (P(n))$.
  So using course of values induction we have shown that

$$P(0) \wedge \forall k \in \mathbb{N}\ (P(k) \to P(k{+}1))\ \to\ \forall n \in \mathbb{N}\ (P(n)),$$

  which states mathematical induction.

- Let P be such that

$$P(0) \tag{4}$$
$$\forall k \in \mathbb{N}\ (\forall i \in \mathbb{N}\ (i \le k \wedge P(i)) \to P(k{+}1)) \tag{5}$$

  We have to show $\forall n \in \mathbb{N}\ (P(n))$. We would like to reach this result using Mathematical Induction, but then we need to show first that

$$\forall k \in \mathbb{N}\ (P(k) \to P(k{+}1)) \tag{6}$$

  Let $Q(n)\ \triangleq\ \forall j \in \mathbb{N}\ (j \le n \wedge P(j))$. We will show $\forall n \in \mathbb{N}\ (Q(n))$, using Mathematical Induction, so we will show

$$\forall n \in \mathbb{N}\ (\forall j \in \mathbb{N}\ (j \le n \wedge P(j))),$$

  so, in particular, $\forall n \in \mathbb{N}\ (P(n))$.

  (*Base case*): $Q(0)\ \triangleq\ \forall j \in \mathbb{N}\ (j \le 0 \wedge P(j)) = P(0)$, so $Q(0)$ holds by assumption (4).

  (*Inductive case*): To prove $Q(k) \to Q(k{+}1)$, for all k, first assume $Q(k)$ so assume

$$\forall j \in \mathbb{N}\ (j \le k \wedge P(j)).$$

  Then, by assumption (5), we also have $P(k{+}1)$, and therefore

$$\forall j \in \mathbb{N}\ (j \le k{+}1 \wedge P(j))$$

  holds, which corresponds to $Q(k{+}1)$.

  So $\forall n \in \mathbb{N}\ (Q(n))$ follows by mathematical induction. This is defined as $\forall j \in \mathbb{N}\ (j \le n \wedge P(j))$, so in particular also $\forall n \in \mathbb{N}\ (P(n))$.
  So using mathematical induction we have shown that

$$P(0) \wedge \forall k \in \mathbb{N}\ (\forall i \in \mathbb{N}\ (i \le k \wedge P(i)) \to P(k{+}1))\ \to\ \forall n \in \mathbb{N}\ (P(n)),$$

  which states the principle of complete induction.

Because of this result, we are free to choose which induction technique we want to apply. The choice is normally decided by the character of the property we try to show.

9.3 Structural induction

In general, especially in the context of computer science, we will define many sets, relations, *etc.*, as the *least* ones satisfying a set of conditions or rules. Normally these definitions are *incremental*, in the sense that we define a set by first specifying the basic elements it contains, and then how new elements can be added by specifying how to create them out of existing elements. We also call these definitions *structural*.

Definition 9.6 (STRUCTURAL DEFINITION) A set V is said to be defined by *structural induction* when it is defined as follows:

(*Base case*): Some *constants* $a_1, \ldots, a_n$ are assumed to be in V.

(*Inductive case*[17]): There is a limited (finite) number of *contexts*[18] $C_1^{n_1}, \ldots, C_m^{n_m}$, also called *constructors*, that, given a number of elements of V, produce another element of V:

$$\text{if } e_1, \ldots, e_{n_1} \in V, \text{ then } C_1^{n_1}[\, e_1, \ldots, e_{n_1} \,] \in V.$$
$$\vdots$$
$$\text{if } e_1, \ldots, e_{n_m} \in V, \text{ then } C_m^{n_m}[\, e_1, \ldots, e_{n_m} \,] \in V.$$

The e_i are called *subexpressions* (or *subterms*, depending on the preferred terminology), of the expression $C^n[\, e_1, \ldots, e_n \,]$.

(*Closure*): Let W be such that $a_i \in W$ for every constant a_i, and that if $e_1, \ldots, e_{n_i} \in W$, then $C_1^{n_i} i[e_1, \ldots, e_{n_i}] \in W$ for every constructor $C_i^{n_i}$, then $V \subseteq W$.

Notice that the use of the ellipses is a short-hand; we can only write down the set if we know the exact number m of constructors and how many subexpressions they need. The closure clause case expresses that V is the smallest set satisfying the first two cases; it is there to make sure that there are no elements in V that are not either one of the listed constants or constructed using the inductive step.

It is common to not separate *constants* from *constructors*, but rather to treat constants as constructors of *arity* zero, and deal with both constants and constructors in *one* go. We will use both expressions, whenever convenient.

We give some examples:

Example 9.7 *1*) We can reformulate the definition of natural numbers:

(*Base case*): $0 \in I\!N$.

(*Inductive Case*): If $n \in I\!N$, then $Succ(n) \in I\!N$.

There is only one constant, 0, and the only constructor is $Succ(\cdot)$.

2) The set of all (finite) lists over natural numbers, $List(I\!N)$, is defined structurally by:

(*Base case*): $[\,] \in List(I\!N)$.

(*Inductive Case*): If $n \in I\!N$, and $l \in List(I\!N)$, then $Cons(n,l) \in List(I\!N)$.

The only constant is $[\,]$, and there is a constructor $Cons(n,\cdot)$ for every $n \in I\!N$; we often write $n : l$ for $Cons(n,l)$.

3) The set of all binary trees over natural numbers is defined by $Tree(I\!N)$, is defined structurally by:

(*Base case*): $n \in Tree(I\!N)$.

(*Inductive Case*): If $t_1, t_2 \in Tree(I\!N)$, then $\langle t_1, t_2 \rangle \in Tree(I\!N)$.

There are infinitely many constants, and only one constructor, $\langle \cdot, \cdot \rangle$.

Let V be an inductively defined set. We now can prove a property by induction using this definition, after we associate the notion of *size* to each expression. For example, we can define

1) For numbers:

[17] Often also called the *recursive case*; since recursion need not terminate, we prefer our terminology.

[18] A context C^n is a term with n 'open' positions, normally represented by '$\cdot$' and we write $C^n[\, e_1, \ldots, e_n \,]$ for the expression obtained from that term by replacing the open positions with the expressions $e_1, \ldots, e_n$.

$$\begin{aligned} size(0) &= 0 \\ size(Succ(n)) &= size(n)+1 \end{aligned}$$

2) For lists:

$$\begin{aligned} size([\,]) &= 0 \\ size(Cons(n,l)) &= size(l)+n+1 \end{aligned}$$

3) For trees:

$$\begin{aligned} size(n) &= 1 \\ size(\langle n_1,n_2\rangle) &= \max(size(t_1),size(t_2))+1 \end{aligned}$$

We only need to make sure that we can guarantee that $size(e_i) < size(\mathrm{C}_i^{n_i}[e_1,\ldots,e_{n_i}])$, for every subexpression e_i (hence the '$+1$' in the second definition).

Now, when trying to show properties over these sets, we can use induction over the size of an expression. We can use mathematical induction for the first case, but that would not be possible for the other two. Rather, there we would need complete induction, since the size of, for example, n_1 and n_2 in the last case would not be 1 less than that of $\langle n_1,n_2\rangle$, but some number smaller than that.

But since it will always be possible to define such a mapping *size*, for any structural definition, we can always show properties using (complete) induction. Such a proof would reason like:

Theorem *$P(e)$ holds for all expressions e in an structurally defined set V.*

Proof: To prove $P(e)$, we use complete induction on the size of expressions.

(*Base case*): Show $P(a_i)$ for all constants in V.

(*Inductive case*): To prove $P(\mathrm{C}_i^{n_i}[e_1,\ldots,e_{n_i}])$, first calculate $size(\mathrm{C}_i^{n_i}[e_1,\ldots,e_{n_i}])$, and observe that, by definition,

$$size(e_j) < size(\mathrm{C}_i^{n_i}[e_1,\ldots,e_{n_i}])$$

for all $j \in \{0,\ldots,n_i\}$. Thereby we can assume $P(e_j)$, for all $j \in \{0,\ldots,n_i\}$. Using this information, we show $P(\mathrm{C}_1^{n_1}[e_1,\ldots,e_{n_1}])$.

We do this for all constructors.

So, by mathematical induction, $P(e)$ holds for all expressions $e \in V$.

Notice that we have abstracted away from the problem how difficult it is to show either of these things, but just assume it is possible.

Using this result, generalising Peano's principle, because of the closure condition we can also see V itself as 'having' an induction principle, and abbreviate the above proof using structural induction:

Definition 9.8 (STRUCTURAL INDUCTION) To prove $P(e)$ for all expressions in a set V defined structurally, it suffices to:

(*Base case*): Show $P(a_i)$ for all constants in V.

(*Inductive case*): For every constructor C^n, we show $P(\mathrm{C}^n[e_1,\ldots,e_n])$, where we can assume $P(e_i)$, for all $i \in \{0,\ldots,n\}$, if needed.

Then $P(e)$, for all expressions $e \in V$, follows by *structural induction*.

The proof follows the structure of expressions, assuming that the property to show holds for smaller expressions, and is therefore known as a *proof by structural induction*. We are allowed to do this, since we can map structural induction to complete induction.

In a sense, when in mathematical induction we use $Succ(n)$ rather than $n+1$, it is easy to see that mathematical induction is structural induction as well. It is easy to check that, apart from the axioms about equality, Definition 3.1 is an instance of the above. Moreover, observe that the closure condition of Definition 9.6 implies the principle of structural induction, and vice versa.

9.4 Grammars

A common way to define a set structually is through a *grammar*; this is a set of rules, specifying how to create elements of the sets (or sentences in a language) in steps. We will use a simplified version of the Backus-Naur form to specify grammars, a notation technique for context-free grammars.

Definition 9.9 (BNF GRAMMAR) *1*) A BNF grammar defines 'classes' whose names are written in angle brackets. A BNF specification is a set of derivation rules, written as

$$\langle\texttt{symbol}\rangle ::= \texttt{\_\_expression\_\_}$$

where the '$\texttt{\_\_expression\_\_}$' consists of one or more sequences of symbols, separated by the vertical bar '$|$', indicating a choice, the whole being a possible substitution for the symbol on the left.

2) Symbols that never appear on a left-hand side are *terminals*. On the other hand, symbols that appear on a left-hand side are *non-terminals* and are always enclosed between the brackets '$\langle$' and '$\rangle$'.

3) The '::=' means that the symbol on the left must stand for one of the choices in the expression on the right; if that contains occurrences of non-terminals, those need to be instantiated using their definitions as well; only expressions without non-terminals are considered to be 'generated by the grammar'.

4) A grammar implicitly defines a language as the smallest set of expressions that can be generated from the grammar.

A grammar specifies how to create a set of 'sentences' of a language (of which this is the grammar).

Example 9.10 An example of such a grammar in BNF is:

$$\begin{aligned}
\langle\texttt{expr}\rangle &::= \langle\texttt{num}\rangle \mid \langle\texttt{expr}\rangle\langle\texttt{operation}\rangle\langle\texttt{expr}\rangle \\
\langle\texttt{operation}\rangle &::= + \mid - \mid \times \\
\langle\texttt{num}\rangle &::= \langle\texttt{digit}\rangle \mid \langle\texttt{digit}\rangle\langle\texttt{num}\rangle \\
\langle\texttt{digit}\rangle &::= 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9
\end{aligned}$$

This grammar can be used to generate a set of expressions built out of numbers, '$+$', '$-$', and '$\times$', like $\{1,\ 27+3,\ 112\times 6780,\ 7+13\times 56,\ \ldots\}$. Notice that each occurrence of $\langle\texttt{expr}\rangle$ is eventually replaced with an expression, and that the two occurrences of $\langle\texttt{expr}\rangle$ in $\langle\texttt{expr}\rangle + \langle\texttt{expr}\rangle$ need not be replaced by the same. We have four non-terminals, being `expr`, `num`, `operation`, and `digit`, and terminal symbols $+$, $-$, $\times$, 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9.

This grammar can be understood as: an expression is either a number, or an expression followed by an operation and followed by an expression; the assumption is that these two expressions are elements of the language so are made using the grammar as well. An operation is either a plus, a minus, or a times symbol, and a number is a digit or a digit followed by a number, where a digit is one of the symbols 0 to 9.

When creating an expression in this language, we have two options. Either we choose it to be a 'number', or that it is a an 'expression', followed by an 'operation', followed by an 'expression'. In the latter case, we have then to build the words that takes the position of the 'expression's, etc, until we stop by choosing the first alternative. The implied assumption is that this process is finite; we do not let a grammar specify objects of infinitely large size.

Grammars are essential for programming languages. Not only do they specify exactly how a program should be constructed, but also allow for the specification of *parsers*, programs that can check if a supplied text can be accepted by checking if it can be generated from the grammar. These parsers are invariably part of the first stage of the compilation of code, and can only be defined on a well-defined grammar.

The brackets '$\langle$' and '$\rangle$' are handy when the name of the class is not a simple symbol, as in Example 9.10. Another common way of writing grammars is to use words (so without spaces) for the non-terminals and to omit the brackets '$\langle$' and '$\rangle$', and surround terminal symbols with '`' and '''. For the above example, this would give:

$$\begin{aligned}\texttt{expr} &::= \texttt{num} \mid \texttt{expr operation expr}\\ \texttt{operation} &::= \text{'}+\text{'} \mid \text{'}-\text{'} \mid \text{'}\times\text{'}\\ \texttt{num} &::= \texttt{digit} \mid \texttt{digit num}\\ \texttt{digit} &::= \text{'0'} \mid \text{'1'} \mid \text{'2'} \mid \text{'3'} \mid \text{'4'} \mid \text{'5'} \mid \text{'6'} \mid \text{'7'} \mid \text{'8'} \mid \text{'9'}\end{aligned}$$

Often non-terminals are used that consist of a single character. Also, to stress that the occurrences of non-terminals represent different terms, we can use indices or will allow us to use more than one non-terminal on the left. Notice that this grammar now creates expressions like '$1+1\times 1$', where the priority of the operations is not clear. It would in this case be better to add parentheses; combining these two observations gives:

$$\begin{aligned}e,f &::= n \mid (e \circ f)\\ \circ &::= + \mid - \mid \times\\ n &::= d \mid d\,n\\ d &::= 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9\end{aligned}$$

This grammar now produces expressions of the shape 1, $(27+3)$, (112×6780), $((7+13)\times 56)$, $(7+(13\times 56))$, ….

We can abstract a little from the actual description of numbers (but of course cannot do that when dealing with a programming language) and use an *abstract syntax*:

$$\begin{aligned}e,f &::= n \mid (e \circ f)\\ \circ &::= + \mid - \mid \times\end{aligned}$$

Notice that we have left the syntax of numbers unspecified.

Example 9.11 *1*) We can define the set V of Definition 9.26 using a grammar:

$$t ::= a_1 \mid \cdots \mid a_n \mid C_1^{n_1}[t_1,\ldots,t_{n_1}] \mid \cdots \mid C_n^{n_m}[t_1,\ldots,t_{n_m}]$$

2) We can define the set of natural numbers through the grammar:

$$n ::= 0 \mid Succ(n)$$

3) The set of all (finite) lists over natural numbers, $List(\mathbb{N})$, is defined through the grammar:

$$\begin{aligned}L &::= [\,] \mid Cons(n,L)\\ n &::= 0 \mid Succ(n)\end{aligned}$$

or, in a more programming-language style:

$$\begin{aligned}L &::= [\,] \mid Cons(n,L)\\ n &::= d \mid d\,n\\ d &::= 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9\end{aligned}$$

This now is completely equivalent to Definition 9.6, so can be preferred since taking up a lot less ink to write down. As with the structural definition of V, we can only write down the grammar if we know the exact number (m) of constructors C_j and their arity n_j.

So grammars correspond to structural definitions; therefore it is not necessary to state a separate principle of induction for them and we can extend our notion of structural induction to sets defined through grammars. This implies that when looking to prove a property for all the expressions generated by a grammar, it suffices to show the case for all the terminal symbols, and that, assuming it holds for the sub-expressions, we can show that the property holds for the composite ones.

Example 9.12 We define the set $\mathcal{S}$ of *strings* over the alphabet $\Sigma = \{\texttt{a},\texttt{b},\texttt{c},\ldots,\texttt{z}\}$, ranged over by c, through the grammar:

$$s ::= \epsilon \mid \langle c \cdot s \rangle$$

where ϵ represents the *empty string* (often λ or even Λ is used for ϵ). Normally the string constructor '$\cdot$' is omitted, but we write it for clarity.

Concatenation of strings is defined through

$$\begin{array}{rcl} \epsilon \circ s_1 & \triangleq & s_1 \\ \langle c \cdot s_1 \rangle \circ s_2 & \triangleq & \langle c \cdot s_1 \circ s_2 \rangle \end{array}$$

We define the notion of *length* of a string as follows:

$$\begin{array}{rcl} \mathit{length}(\epsilon) & \triangleq & 0 \\ \mathit{length}(\langle c \cdot s \rangle) & \triangleq & 1 + \mathit{length}(s) \end{array}$$

We define the notion of *reverse* of a string inductively as follows:

$$\begin{array}{rcl} \mathit{rev}(\epsilon) & \triangleq & \epsilon \\ \mathit{rev}(\langle c \cdot s \rangle) & \triangleq & \mathit{rev}(s) \circ \langle c \cdot \epsilon \rangle \end{array}$$

Since the functions defined here follow the structure of the terms, we can consider them also *defined structurally*.

Notice that we have used an abstract grammar to present strings. A full grammar would have been:

$$\begin{array}{rcl} s & ::= & \epsilon \mid \langle c \cdot s \rangle \\ c & ::= & \mathtt{a} \mid \mathtt{b} \mid \mathtt{c} \mid \mathtt{d} \mid \mathtt{e} \mid \mathtt{f} \mid \mathtt{g} \mid \mathtt{h} \mid \mathtt{i} \mid \mathtt{j} \mid \mathtt{k} \mid \mathtt{l} \mid \mathtt{m} \mid \mathtt{n} \mid \mathtt{o} \mid \mathtt{p} \mid \mathtt{q} \mid \mathtt{r} \mid \mathtt{s} \mid \mathtt{t} \mid \mathtt{u} \mid \mathtt{v} \mid \mathtt{w} \mid \mathtt{x} \mid \mathtt{y} \mid \mathtt{z} \end{array}$$

We can now use structural induction to show:

Lemma 9.13 For all strings s_1 and s_2:

1) $\mathit{length}(s_1 \circ s_2) = \mathit{length}(s_1) + \mathit{length}(s_2)$.

2) $\mathit{rev}(s_1 \circ s_2) = \mathit{rev}(s_2) \circ \mathit{rev}(s_1)$.

We leave the proof as Exercise 55.

Of course not everything written down using grammar notation is automatically well defined. For example, if in the definition of strings we would omit the empty string ϵ, we are effectively defining *infinite* strings, on which we cannot perform any inductive reasoning, since there is no base case to show.[19]

9.5 Examples of (formal) calculi

We will now look at some examples of grammars for calculi that are studied within theoretical computer science. They are, in a sense, rudimentary programming languages, that are all Turing complete, so powerful enough to express all computable functions.

The λ-calculus

We start with a programming language that embodies the essentials of the functional programming paradigm, Church's *λ-calculus*. It forms the basis for Haskell, and is actually part of it.

The set of *λ-terms* is constructed from an infinite, countable set of term-variables and two operations, *application* and *abstraction*. The BNF-grammar for λ-terms could be written as

$$\begin{array}{rcl} \langle \textsf{lambda term} \rangle & ::= & \langle \textsf{variable} \rangle \mid (\lambda \langle \textsf{variable} \rangle . \langle \textsf{lambda term} \rangle) \mid (\langle \textsf{lambda term} \rangle \, \langle \textsf{lambda term} \rangle) \\ \langle \textsf{variable} \rangle & ::= & x \mid y \mid z \mid \cdots \mid x_1 \mid x_2 \mid \cdots \end{array}$$

but this has as disadvantage that we need to use ellipses to represent the set of variables. Rather, we write:

Definition 9.14 (λ-TERMS) We take $\mathcal{V}$ as the infinite, countable set of term-variables containing lower case characters, possibly indexed, and ranged over by x, y, z, x_1, x_2, etc. We define Λ, the set of λ-terms by the (abstract) grammar:

$$\begin{array}{rcccccc} M, N & ::= & x & \mid & (\lambda x . M) & \mid & (M N) \\ & & \textit{variable} & & \textit{abstraction} & & \textit{application} \end{array}$$

[19] Here the notion of *co-induction* would be more useful.

Church defined his calculus when trying to solve Hilbert's decision problem; he chose to define a language based on the practice of functions in mathematics, but does not assume anything like numbers or such. The operation of application, borrowed from mathematics, takes two terms M and N, and produces a new term, the application of M to N; we can see the term M as a function and N as its operand. The operation of abstraction takes a term-variable x and a term M and produces an abstraction term $(\lambda x.M)$. In a sense, abstraction builds an anonymous, unnamed function; you can read $(\lambda x.M)$ as 'given the operand x, this function returns M'; when naming this function f, we get $f(x) = M$. In the λ-calculus, function definition and application are not separated.

The important notion of *free* and *bound* term-variables of λ-terms is defined by:

Definition 9.15 (FREE AND BOUND VARIABLES) The set of *free* variables of a term M ($fv(M)$) and its *bound* variables ($bv(M)$) are (structurally) defined by:

$$\begin{array}{lcl@{\qquad}lcl} fv(x) &=& \{x\} & bv(x) &=& \emptyset \\ fv(MN) &=& fv(M) \cup fv(N) & bv(MN) &=& bv(M) \cup bv(N) \\ fv(\lambda y.M) &=& fv(M) \setminus \{y\} & bv(\lambda y.M) &=& bv(M) \cup \{y\} \end{array}$$

We will assume that bound and free variables are always different (this is called *Barendregt's convention*).

In modelling calculation using functions, we need the operation of substitution (in mathematics normally not formally specified), to express that the parameter replaces the variable in a function. This feature returns in the λ-calculus and is at the basis of the computational step. The definition of term substitution is:

Definition 9.16 (TERM SUBSTITUTION) The substitution of the term variable x by the term N, written $\{N/x\}$ is defined inductively over the structure of terms by:

$$\begin{array}{rcll} x\{N/x\} &=& N & \\ y\{N/x\} &=& y, & (\textit{if}\ y \neq x) \\ (PQ)\{N/x\} &=& (P\{N/x\}\ Q\{N/x\}) & \\ (\lambda y.M)\{N/x\} &=& (\lambda y.M\{N/x\}) & \end{array}$$

Notice that, since y is bound in $(\lambda y.M)$, by Barendregt's convention it is not free in N, so it is not possible that free occurrences of y in N will be captured by the substitution.

Although defined inductively, by following the grammatical structure of terms, the substitution $\{N/x\}$ is assumed to take place silently, and immediately, and replaces all occurrences of x in parallel. Thereby $(xx)\{N/x\}$ and (NN) are *identical.*

The basic computational step is that of effecting the replacement of a bound variable in an abstraction by the parameter of the application. The notion of computation, called *β-reduction*, is defined as a the transitive closure of the relation '$\rightarrow_\beta$' on terms that specifies that, if $M \rightarrow_\beta N$, then M executes 'in one step' to N.

Definition 9.17 (β-CONVERSION) *1*) The binary *one-step* reduction relation '$\rightarrow_\beta$' on λ-terms is defined by the β-reduction rule (we will write $M \rightarrow_\beta N$ rather than $\langle M,N \rangle \in \rightarrow_\beta$):

$$((\lambda x.M)N) \rightarrow_\beta M\{N/x\}$$

and the 'contextual closure' rules

$$M \rightarrow_\beta N \Rightarrow \left\{ \begin{array}{rcl} (PM) &\rightarrow_\beta& (PN) \\ (MP) &\rightarrow_\beta& (NP) \\ (\lambda x.M) &\rightarrow_\beta& (\lambda x.N) \end{array} \right.$$

2) A term of the shape $(\lambda x.M)N$ is called a *reducible expression* (*redex* for short); the term $M\{N/x\}$ that is obtained by reducing this term is called a *contractum* (from 'contracting the redex') or *reduct.*

3) The relation '$\rightarrow_\beta^*$' (or '$\twoheadrightarrow_\beta$') is defined as the reflexive, transitive closure of '$\rightarrow_\beta$':

$$\begin{array}{c} M \rightarrow_\beta N \ \Rightarrow\ M \rightarrow_\beta^* N \\ M \rightarrow_\beta^* M \\ M \rightarrow_\beta^* N \wedge N \rightarrow_\beta^* P \ \Rightarrow\ M \rightarrow_\beta^* P \end{array}$$

4) '$=_\beta$' is the equivalence relation generated by '$\rightarrow^*_\beta$':

$$\begin{array}{rcl} M \rightarrow^*_\beta N & \Rightarrow & M =_\beta N \\ M =_\beta N & \Rightarrow & N =_\beta M \\ M =_\beta N \wedge N =_\beta P & \Rightarrow & M =_\beta P \end{array}$$

The relation '$\rightarrow^*_\beta$' is often written as '$\twoheadrightarrow_\beta$.'

It is worthwhile to point out that any term can appear in operand position in a redex, and thereby be substituted, including abstractions. This corresponds to the feature of functional programming languages that 'functions are first class citizens': they can be handled as any other object in the language.

Example 9.18 We show some examples of reduction and equality.

Notice that:

$$\begin{array}{ll} ((\lambda x.(\lambda y.(\lambda z.((xz)\,(yz)))))\,(\lambda a.(\lambda b.a))) & \rightarrow_\beta \\ (\lambda y.(\lambda z.(((\lambda a.(\lambda b.a))\,z)\,(yz)))) & \rightarrow_\beta \\ (\lambda y.(\lambda z.((\lambda b.z)\,(yz)))) & \rightarrow_\beta \\ (\lambda y.(\lambda z.z)) & \end{array}$$

$$\begin{array}{ll} (((\lambda u.(\lambda v.(uv)))\,(\lambda c.c))\,(\lambda y.(\lambda z.z))) & \rightarrow_\beta \\ ((\lambda v.((\lambda c.c)\,v))\,(\lambda y.(\lambda z.z))) & \rightarrow_\beta \\ ((\lambda c.c)\,(\lambda y.(\lambda z.z))) & \rightarrow_\beta \\ (\lambda y.(\lambda z.z)) & \end{array}$$

so we have both

$$\begin{array}{rcl} ((\lambda x.(\lambda y.(\lambda z.((xz)\,(yz)))))\,(\lambda a.(\lambda b.a))) & \rightarrow^*_\beta & (\lambda y.(\lambda z.z)) \\ (((\lambda u.(\lambda v.(uv)))\,(\lambda c.c))\,(\lambda y.(\lambda z.z))) & \rightarrow^*_\beta & (\lambda y.(\lambda z.z)) \end{array}$$

and thereby

$$((\lambda x.(\lambda y.(\lambda z.((xz)\,(yz)))))\,(\lambda a.(\lambda b.a))) =_\beta (((\lambda u.(\lambda v.(uv)))\,(\lambda c.c))\,(\lambda y.(\lambda z.z)))$$

Although very compact and simple, the λ-calculus is very powerful, not just as it is Turing complete. Most programming languages, for example, have to add recursion explicitly, but the λ-calculus has specific λ-terms, called *fixed-point combinators*, that can be used to model recursion.[20] Also numbers can be encoded easily, and the partial recursive functions are then easily defined as λ-terms.

One of the main features of the λ-calculus is *confluence*: no matter how you run a (terminating) reduction, it always gives the same outcome.

The π-calculus

Another formalism, that focusses on *processes* and the synchronisation between them, is that of Robin Milner's π-calculus. This is a formalism that has communication between processes as it basic computational step; processes synchronise their execution and exchange data via the *sending* and *receiving* of channel names.

Definition 9.19 (PROCESSES) *1*) *Channel names* are defined by: $a,b,c,\ldots,x,y,z$.

2) Processes are defined by the grammar:

$$\begin{array}{rcll} P,Q & ::= & 0 & (\textit{nil}) \\ & | & P \mid Q & (\textit{composition}) \\ & | & !P & (\textit{replication}) \\ & | & (\nu a)\,P & (\textit{restriction}) \\ & | & a(x)\,.\,P & (\textit{input}) \\ & | & \overline{a}\langle b\rangle\,.\,P & (\textit{output}) \end{array}$$

[20] The story becomes rather different when adding types to the language.

The prefixes '$\overline{a}\langle b\rangle$.' and '$a(x)$.' are called *guards*, and $a(x).P$ binds the occurrences of the name x in P.

3) The *structural congruence* on processes is the smallest congruence[21] generated by the rules:

$$\begin{array}{rcl} P \mid 0 & \equiv & P \\ P \mid Q & \equiv & Q \mid P \\ !P & \equiv & P \mid !P \\ (\nu n)\,0 & \equiv & 0 \end{array} \qquad \begin{array}{rcl} (P \mid Q) \mid R & \equiv & P \mid (Q \mid R) \\ (\nu m)\,(\nu n)\,P & \equiv & (\nu n)\,(\nu m)\,P \\ (\nu n)\,(P \mid Q) & \equiv & P \mid (\nu n)\,Q \quad (n \notin fn(P)) \end{array}$$

4) The *reduction relation* over the processes is defined by the following rules:

$$\begin{array}{rcll} \overline{a}\langle b\rangle.P \mid a(x).Q & \to_\pi & P \mid Q[b/x] & (\textit{synchronisation}) \\ P \to_\pi P' & \Rightarrow & (\nu n)\,P \to_\pi (\nu n)\,P' & (\textit{hiding}) \\ P \to_\pi P' & \Rightarrow & P \mid Q \to_\pi P' \mid Q & (\textit{composition}) \\ P \equiv Q \wedge Q \to_\pi Q' \wedge Q' \equiv P' & \Rightarrow & P \to_\pi P' & (\textit{structural congruence}) \end{array}$$

Reduction in the π-calculus is not confluent, which makes reasoning about it rather complicated.

The π-calculus has some features that are not found in other programming paradigms. Unlike the λ-calculus, it is first-order, since only channel names can be passed between processes, not processes themselves.[22] Also, reduction is not allowed under guard, so is not defined as generally as in the λ-calculus. As shown by Davide Sangiorgi, it is possible to encode the λ-calculus into the π-calculus, but not the full reduction; it is possible to show that the full meaning of the terms is preserved, though.

Another difference is that the unfolding of replication $!P$ via the congruence $!P \equiv P \mid !P$, used to model recursion, is not a computational step, but rather a purely syntactic operation. Some variants of the calculus deal with this by restricting the structural congruence relation, effectively removing $!P \equiv P \mid !P$, but adding a rule that only allows the '!' to develop if the process underneath is ready to input on a channel 'a', and in parallel to a process that is ready to output on that same channel, as in:

$$!a(y).P \mid (\nu b)\,(\overline{a}\langle b\rangle.R \mid Q) \;\to_\pi\; !a(y).P \mid (\nu b)\,(P\{a/y\} \mid R \mid Q)$$

Featherweight Java

Another programming paradigm is that of *object orientation*, where the distinction is between *object* and *class*-based languages. To study the fundamental properties of these languages, Atsushi Igarashi, Benjamin Pierce, and Philip Wadler defined *Featherweight Java*, a Turing complete class-based language with methods, fields, class definitions and class extensions with method override.

As in other class-based object-oriented languages, FJ's *classes* represent abstractions encapsulating both data (stored in *fields*) and the operations to be performed on that data (encoded as *methods*). Sharing of behaviour is accomplished through the *inheritance* of fields and methods from parent classes. Computation is mediated by *instances* of these classes (called *objects*), which interact with one another by *calling* (also called *invoking*) methods on each other and accessing each other's (or their own) fields.

We do not consider cast expressions since, as the creators of FJ themselves point out, the presence of *downcasts* is unsound[23]; for this reason we present our calculus FJ here without casts.

As is usual, we distinguish the class name `Object` (which denotes the root of the class inheritance hierarchy in all programs) and the self variable `this`[24] used to refer to the receiver object in method bodies. When a method is called, its parameters will take the positions of the variables in the method body, comparable to substitution in the λ-calculus. The self variable will be replaced by the object that does the method invocation; this is the object-oriented way of achieving recursion.

[21] A *pre-congruence* is a reflexive and transitive relation that is preserved in all contexts; a *congruence* is symmetric pre-congruence.

[22] This is very different from the Higher Order π-calculus, or the Ambient Calculus.

[23] In the sense that typeable expressions can get stuck at runtime.

[24] Not a variable in the traditional sense, since it is not used to express a position in the method's body where a parameter can be passed.

Definition 9.20 (FJ$^{\not{c}}$ SYNTAX) A FJ$^{\not{c}}$ program *Prog* consist of a *class table* $\mathcal{CT}$, comprising the *class declarations*, and an *expression* $\mathtt{e}$ to be run (corresponding to the body of the `main` method in a real Java program). They are defined by the grammar:

$$\begin{array}{rcll}
\mathtt{e} & ::= & x \mid \mathtt{this} \mid \mathtt{new}\ C(\vec{\mathtt{e}}) \mid \mathtt{e}.f \mid \mathtt{e}.m(\vec{\mathtt{e}})\ ^{25} & \\
fd & ::= & C\,f; & \\
md & ::= & D\ m(C_1\ x_1,\ \ldots,\ C_n\ x_n)\ \{\mathtt{return}\ \mathtt{e};\} & \\
cd & ::= & \mathtt{class}\ C\ \mathtt{extends}\ C'\ \{\overrightarrow{fd}\ \ \overrightarrow{md}\} & (C \neq \mathtt{Object}) \\
\mathcal{CT} & ::= & \overrightarrow{cd} & \\
Prog & ::= & (\mathcal{CT},\mathtt{e}) &
\end{array}$$

Notice that, contrary to λ-calculus, the language is first-order, and that there is no notion of application. Moreover, observe that a class C can *extend* another D; then C will *inherit* all methods and fields from D and C is a *sub-class* of D.

From this point, for readability, all the concepts defined are program dependent (or more precisely, parametric on the class table); however, since a program is essentially a fixed entity, the class table will be left as an implicit parameter in the definitions that follow. We also only consider programs which conform to some sensible well-formedness criteria: no cycles in the inheritance hierarchy, and fields and methods in any given branch of the inheritance hierarchy are uniquely named. An exception is made to allow the redeclaration of methods, providing that only the *body* of the method (so not the types) is allowed to differ from the previous declaration (*method override*).

Because a class can be defined as an extension of another class, not all fields and methods that are available to this (new) class are present in its definition; some might be only defined in classes higher in the class hierarchy. We therefore define the following functions to look up elements of class definitions.

Definition 9.21 (LOOKUP FUNCTIONS) The following lookup functions are defined to extract the names of fields and bodies of methods belonging to (and inherited by) a class.

1) The following functions retrieve the name of a class or field from its definition:

$$\begin{array}{ll}
\textsc{cn}(\mathtt{class}\ C\ \mathtt{extends}\ D\ \{\overrightarrow{fd}\ \ \overrightarrow{md}\}) & = C \\
\textsc{fn}(C\,f) & = f
\end{array}$$

2) By abuse of notation, we will treat the *class table*, $\mathcal{CT}$, as a partial map from class names to class definitions:

$$\mathcal{CT}(C)\ =\ cd\ \text{ if } \textsc{cn}(cd) = C \text{ and } cd \in \mathcal{CT}$$

3) The list of fields belonging to a class C (including those it inherits) is given by the function $\mathcal{F}$, which is defined as follows:

$$\begin{array}{rcl}
\mathcal{F}(\mathtt{Object}) & = & \epsilon \\
\mathcal{F}(C) & = & \mathcal{F}(C')\cdot\vec{f}\ \text{ if } \mathcal{CT}(C) = \mathtt{class}\ C\ \mathtt{extends}\ C'\ \{\overrightarrow{fd}\ \ \overrightarrow{md}\} \\
& & \text{and } \textsc{fn}(fd_i) = f_i \text{ for all } i \in \overline{n}.
\end{array}$$

4) The function *Mb*, given a class name C and method name m, returns a tuple $(\vec{x},\mathtt{e})$, consisting of a sequence of the method's formal parameters and its body: $\{\mathtt{a}\}\{a\}$

$$\begin{array}{rcl}
Mb(C,m) & = & (\vec{x},\mathtt{e}) \\
& & \text{if } \mathcal{CT}(C) = \mathtt{class}\ C\ \mathtt{extends}\ C'\ \{\overrightarrow{fd}\ \ \overrightarrow{md}\}\text{, and there exist} \\
& & C_0,\vec{C} \text{ such that } C_0\ m(C_1\,x_1,\ldots,C_n\,x_n)\ \{\mathtt{return\ e;}\} \in \overrightarrow{md}. \\[1em]
Mb(C,m) & = & Mb(C',m) \\
& & \text{if } \mathcal{CT}(C) = \mathtt{class}\ C\ \mathtt{extends}\ C'\ \{\overrightarrow{fd}\ \ \overrightarrow{md}\}\text{, and there are no} \\
& & C_0,\ \vec{C},\ \vec{x},\ \mathtt{e} \text{ such that } C_0\ m(C_1\,x_1,\ldots,C_n\,x_n)\ \{\mathtt{return\ e;}\} \in \overrightarrow{md}.
\end{array}$$

5) $\textsc{vars}(\mathtt{e})$ returns the set of variables used in the expression $\mathtt{e}$.

We impose the additional criterion that well-formed programs satisfy the following property:

[25] We use the vector notation $\vec{\cdot}$ for a number of occurrences of the symbol underneath; so $\vec{\mathtt{e}}$ stands for $\mathtt{e}_1,\ldots,\mathtt{e}_n$ for some n.

$$\text{if } Mb(\mathtt{C},m) = (\vec{x},\mathtt{e}_{\mathrm{b}}) \text{ then } \textsc{vars}(\mathtt{e}_{\mathrm{b}}) \subseteq \{x_1,\dots,x_n\}.$$

As is clear from the definition of classes, if `class` C `extends` D $\{\vec{f}\ \overline{m}\}$ is a class declaration, then `new` $C(\vec{e})$ creates an object where the expression e_i is attached to the field f_i.

As for the λ-calculus, *substitution* of expressions for variables is the basic mechanism for reduction in FJ: when a method is invoked on an object (the *receiver*) the invocation is replaced by the body of the method that is called, and each of the variables is replaced by a corresponding argument, and `this` by the receiver.

Definition 9.22 (REDUCTION) The reduction relation $\rightarrow$ is defined by:

$$\begin{array}{rcl} \mathtt{new}\ C(\vec{e})\,.f_i & \rightarrow & e_i \quad \text{for class name } C \text{ with } \mathcal{F}(C) = \vec{f} \text{ and } i \in \overline{n}, \\ \mathtt{new}\ C(\overrightarrow{e'})\,.m(\overrightarrow{e_n}) & \rightarrow & e^{\mathsf{S}} \quad \text{for class name } C \text{ and method } m \text{ with } Mb(C,m) = (\vec{x},e'), \\ & & \qquad \text{where } \mathsf{S} = \{\ \mathtt{this} \mapsto \mathtt{new}\ C(\overrightarrow{e'}),\ \mathsf{x}_1 \mapsto e_1,\ \dots,\ \mathsf{x}_n \mapsto e_n\ \} \end{array}$$

We add the usual congruence rules for allowing reduction in subexpressions, and the reflexive and transitive closure of $\rightarrow$ is denoted by $\rightarrow^*$.

The language **`While`**

To conclude our examples, for completeness we will quickly show the language `While`. It defines *arithmetic* and *boolean* expressions, as well as *programs*, built using a *assignment* of the result of an arithmetic expression to a *variable*, *program composition* through juxtaposition, and a *conditional* and a *loop construct*.

Example 9.23 (THE LANGUAGE `WHILE`) The abstract syntax for the basic programming language `While` is given by:

$$\begin{array}{rcl} a & ::= & n \mid x \mid (a_1 \circ a_2) \\ \circ & ::= & + \mid - \mid \times \\ b & ::= & \mathtt{true} \mid \mathtt{false} \mid (a_1 = a_2) \mid (a_1 \leq a_2) \mid (\neg b) \mid (b_1 \wedge b_2) \\ P & ::= & x := a \mid \mathtt{skip} \mid (P_1\,;\,P_2) \mid (\mathtt{while}\ b\ \mathrm{do}\ P) \mid (\mathtt{if}\ b\ \mathtt{then}\ P_1\ \mathtt{else}\ P_2) \end{array}$$

This programming language is popular when studying elementary properties of imperative programming languages. Since the language has variables and *assignment* it is a language with *side effects* and needs a representation of the current state of the memory to express how programs are running, so we cannot give a notion of reduction on terms to indicate how programs run, as we did for the three calculi before.

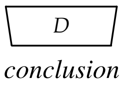
9.6 Inference rules

Another very common way of presenting a definition is to use *inference rules*, that have the generic shape:

$$(\mathit{name}) : \frac{\mathit{premises}}{\mathit{conclusion}}$$

with the intended meaning: '*if all the premises hold, then so does the conclusion.*'; sometimes in presenting the rule the name is omitted.

These rules can then be applied repeatedly, letting the conclusions of one rule be the premises of the next, to form what are called *derivations*, which we represent with boxes (perhaps with a name in it) with a conclusion underneath:



For readability we sometimes put the name of the rule applied to the right of the step.

A collection of inference rules normally has some members that have no premises, that will then form the leaves of the derivations.

As with inductive definitions, care is needed. What set is defined by:

$$\frac{n \in X}{n+3 \in X}$$

or:

$$\frac{}{0 \in X} \qquad \frac{n \in X}{n+2 \in X} \qquad \frac{n+2 \in X}{n \notin X}$$

We can present many definitions using rules. In fact, every ‘object’ that is defined in terms of ‘if … then … ’ can be formulated using rules. They are a very clean and precise notation for definitions that would otherwise take up a lot of words. However, one important thing to note is that, using systems defined by *rules*, the *nature* of the objects we work with changes. When we want to prove properties of a system that is defined only in terms of ‘*if* … *then* … ’, or through grammars, we would need to follow that structure. When using rules, however, the objects over which you prove become *derivations* constructed by applying the rules.

Example 9.24 We can define the set of λ-terms through:

$$\frac{}{x \in \Lambda}\,(x \in \mathcal{V}) \qquad \frac{M \in \Lambda}{(\lambda x . M) \in \Lambda}\,(x \in \mathcal{V}) \qquad \frac{M \in \Lambda \qquad N \in \Lambda}{(MN) \in \Lambda}$$

Now the relation $M \in \Lambda$ is defined using rules, so any proof on this relation would need to be done ‘by induction on the structure of derivations’.

Example 9.25 We can define the relations statements $M \rightarrow_\beta N$, $M \rightarrow_\beta^* N$ and $M =_\beta N$ for the λ-calculus through the rules:

$$\frac{}{(\lambda x . M)\, N \rightarrow_\beta M\{N/x\}} \qquad \frac{M \rightarrow_\beta N}{MP \rightarrow_\beta NP} \qquad \frac{M \rightarrow_\beta N}{PM \rightarrow_\beta PN} \qquad \frac{M \rightarrow_\beta N}{\lambda x . M \rightarrow_\beta \lambda x . N}$$

$$\frac{M \rightarrow_\beta N}{M \rightarrow_\beta^* N} \qquad \frac{}{M \rightarrow_\beta^* M} \qquad \frac{M \rightarrow_\beta^* N \quad N \rightarrow_\beta^* P}{M \rightarrow_\beta^* P} \qquad \frac{M \rightarrow_\beta^* N}{M =_\beta N} \qquad \frac{M =_\beta N}{N =_\beta M} \qquad \frac{M =_\beta N \quad N =_\beta P}{M =_\beta P}$$

It is thereby possible to see the definition of these relations in Definition 9.17 as inductive as well.

9.7 Inductive definitions

So far we have looked at definitions of sets that are syntactic or have significant structure, in the sense that they are defined *inductively* using constructors that build new elements.

But there is an even more generic approach to inductive definitions.

Definition 9.26 (INDUCTIVE DEFINITION) A set V is said to be *inductively defined* when its definition follows the structure:

1) Some elements are in V as given;

2) If these elements are in V, then so are those.

3) Only elements selected like this are in V.

We have seen definitions like these above, notably in

- Definition 9.17, where the sets defined inductively are, respectively ‘$\rightarrow_\beta$’, ‘$\rightarrow_\beta^*$’ and ‘$=_\beta$’ (as binary relations on Λ) all are subsets of Λ^2 that are defined inductively.
- Definition 9.19, where the relations ‘$\rightarrow_\pi$’ and ‘$\equiv$’ are defined inductively.

Notice that, according to our criterion (Definition 9.6), those relations are not defined using structural induction. Defining the appropriate notion of length of an element, using the number of steps needed to create it in the definition to come to this element, we can show properties over those relation using mathematical induction; as before, we can therefore show properties over *the inductive definition* of the properties.

For example, we can show that, when $M =_\beta N$, then there are terms M_1, M_2, …, M_n, M_{n+1} such that $M \equiv M+1$, $N \equiv M_{n+1}$, and, for all $1 \leq i \leq n$, either $M_i \rightarrow_\beta^* M_{i+1}$, or $M_{i+1} \rightarrow_\beta^* M_i$ by induction on the definition of '$=_\beta$'. We leave this as Exercise 59.

* 9.8 Natural deduction for Implicative Logic

Rules and derivations are the common tool to present logic and proofs.

Definition 9.27 *1*) We define *implicative logical formulas* through the grammar:

$$A,B ::= \top \mid \bot \mid (A \to B)$$

We normally only use those brackets in formulas that are needed to avoid ambiguity.

2) In Implicative Logic (IL) we derive *statements* (or *judgements*) of the shape $\Gamma \vdash A$, where Γ is a set of logical formulas, the *assumptions*, and A is the *conclusion*. We can read $\Gamma \vdash A$ as 'Γ *shows* A', or '*from* Γ *we can deduce* A', or 'A *follows from the assumptions in* Γ.

3) *Proofs in* IL are defined as the smallest set of derivations that can be generated from the inference rules:

$$(Ax): \frac{}{\Gamma \cup \{A\} \vdash A} \qquad (\to I): \frac{\Gamma \cup \{A\} \vdash B}{\Gamma \vdash A \to B} \qquad (\to E): \frac{\Gamma \vdash A \to B \quad \Gamma \vdash A}{\Gamma \vdash B}$$

4) We write $\mathcal{D} :: \Gamma \vdash A$ if there exists a derivation $\mathcal{D}$ built using the rules that has the judgement $\Gamma \vdash A$ as conclusion, and say that that $\mathcal{D}$ shows $\Gamma \vdash A$. We write $\Gamma \vdash A$ when there exists a derivation $\mathcal{D}$ such that $\mathcal{D} :: \Gamma \vdash A$.

It is straightforward to add the rules for the other logical connectives.

We can write rule (Ax) also as

$$\frac{}{\Gamma \vdash A}\ (A \in \Gamma)$$

Notice that we have not given a definition of statements of the shape $\Gamma \vdash A$, but *of derivations that show those*. This implies that we never reason over the structure of the judgements, but always over the structure of derivations.

The first inference rule (in this context also called a *proof rule*) (Ax), states that, for any set Γ of formulae, and for any A,

$$\frac{}{\Gamma \cup \{A\} \vdash A}\ (Ax)$$

is a correct derivation. Moreover, suppose we have constructed a derivation D that *ends with* (or *derives*) $\Gamma \cup \{A\} \vdash B$, *i.e*

$$\begin{array}{c} \boxed{\quad D \quad} \\ \Gamma \cup \{A\} \vdash B \end{array}$$

then by the second rule $(\to I)$, we can create a correct derivation when we draw a line underneath this, and put the line $\Gamma \vdash A \to B$ underneath that line.

$$\begin{array}{c} \boxed{\quad D \quad} \\ \dfrac{\Gamma \cup \{A\} \vdash B}{\Gamma \vdash A \to B}\ (\to I) \end{array}$$

(Notice that A has moved from left to right with respect to the turnstile.)

And, similarly, if we have two separate derivations D_1 and D_2, one with last line $\Gamma \vdash A \to B$, the other with last line $\Gamma \vdash A$, then by the last rule $(\to E)$, when we draw a line underneath them both, and put the line $\Gamma \vdash B$ under it all, we obtain a correct derivation:

$$\dfrac{\begin{array}{c} \boxed{\quad D_1 \quad} \\ \Gamma \vdash A \to B \end{array} \qquad \begin{array}{c} \boxed{\ D_2\ } \\ \Gamma \vdash A \end{array}}{\Gamma \vdash B}\ (\to E)$$

These three steps are the only ones permitted to construct derivations in IL; in other words: the *set of derivations for* IL *is the smallest set closed for the three* derivation constructors, *the rules* (Ax), $(\to I)$, *and* $(\to E)$ *given above*.

In short:

1) For any Γ and $A \in \Gamma$, there exists a derivation that shows $\Gamma \vdash A$.

2) For any Γ, and A and B, if there exists a derivation that shows $\Gamma \cup \{A\} \vdash B$, then there exists a derivation that shows $\Gamma \vdash A \to B$.

3) For any Γ, and A and B, if there exists a derivation that shows $\Gamma \vdash A \to B$ and there exists a derivation that shows $\Gamma \vdash A$, then there exists a derivation that shows $\Gamma \vdash B$.

Since the words 'there exists a derivation that shows' are implied by Definition 9.27 (*4*), we get:

1) For any Γ and $A \in \Gamma$, $\Gamma \vdash A$.

2) For any Γ, and A and B, if $\Gamma \cup \{A\} \vdash B$, then $\Gamma \vdash A \to B$.

3) For any Γ, and A and B, if $\Gamma \vdash A \to B$ and $\Gamma \vdash A$, then $\Gamma \vdash B$.

Example 9.28 Let $\Gamma = \{A \to B \to A, A \to B, A\}$; the following is an example of a proof in IL:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash A \to B \to A}\,(Ax) \quad \dfrac{}{\Gamma \vdash A}\,(Ax)}{\Gamma \vdash B \to A}\,(\to E) \qquad \dfrac{\dfrac{}{\Gamma \vdash A \to B}\,(Ax) \quad \dfrac{}{\Gamma \vdash A}\,(Ax)}{\Gamma \vdash B}\,(\to E)}{\Gamma \vdash A}\,(\to E)}{\{A \to B \to A, A \to B\} \vdash A \to A}\,(\to I)}{\{A \to B \to A\} \vdash (A \to B) \to A \to A}\,(\to I)}{\emptyset \vdash (A \to B \to A) \to (A \to B) \to A \to A}\,(\to I) \qquad \dfrac{\dfrac{\dfrac{}{\{A, B\} \vdash A}\,(Ax)}{\{A\} \vdash B \to A}\,(\to I)}{\emptyset \vdash A \to B \to A}\,(\to I)}{\emptyset \vdash (A \to B) \to A \to A}\,(\to E)$$

Notice that the rules we use are actually defining:

Definition 9.29 The set of proofs in IL, $\mathit{prf}(\mathsf{IL})$ is defined by:

1) $\langle Ax \rangle :: \Gamma \cup \{A\} \vdash A \in \mathit{prf}(\mathsf{IL})$.

2) If $\mathcal{D} :: \Gamma \cup \{A\} \vdash B \in \mathit{prf}(\mathsf{IL})$, then $\langle \mathcal{D} :: \Gamma \cup \{A\} \vdash B, \to I \rangle :: \Gamma \vdash A \to B \in \mathit{prf}(\mathsf{IL})$.

3) If $\mathcal{D}_1 :: \Gamma \vdash A \to B \in \mathit{prf}(\mathsf{IL})$ and $\mathcal{D}_2 :: \Gamma \vdash A \in \mathit{prf}(\mathsf{IL})$, then

$$\langle \mathcal{D}_1 :: \Gamma \vdash A \to B, \mathcal{D}_2 :: \Gamma \vdash A, \to E \rangle :: \Gamma \vdash B \in \mathit{prf}(\mathsf{IL}).$$

where Γ is a set of formulas and A and B are formulas.

Notice that here we basically represent the (intended meaning of the) inference rules in linear notation and that this is very much an structural definition, defining proofs (derivations), not just judgements.

Therefore, for any system defined using inference rules we have the full power of structural induction. The *set of derivations* is structurally defined and any (suitable) property over derivable expressions is actually a property over *derivations* that can be proven by *induction on the structure of derivations*.

For IL, the base case would be a derivation consisting (only) of an application of rule (Ax). The inductive cases are then dealing with derivations ending with either rule $(\to I)$ or $(\to E)$; the first has one subderivation for which we can assume, if needed, that the property we have to show holds, and the second has two such sub-derivations.

Example 9.30 Show, for all provable statements $\Gamma \vdash A$ in IL, that property $P(\Gamma \vdash A)$ holds.

Proof: Let $\Gamma \vdash A$ be a provable statement; since this statement stands for 'there exists a derivation $\mathcal{D}$ that shows this result', we show $P(\mathcal{D} :: \Gamma \vdash A)$ by induction on the structure of derivations.

(Ax): (This is the base case of the induction) $\mathcal{D}$ is nothing but an application of rule Ax, so is of the shape:

$$\dfrac{}{\Gamma \cup \{A\} \vdash A}$$

for some Γ and A. We need to show directly that P holds for $\mathcal{D}$.

$(\to I)$: (This is the first inductive case) In this case, we have a derivation $\mathcal{D}'$ such that $\mathcal{D}$ is of the structure

$$\dfrac{\begin{array}{c}\boxed{\quad\mathcal{D}'\quad}\\ \Gamma\cup\{A\}\vdash B\end{array}}{\Gamma\vdash A\to B}$$

The derivation $\mathcal{D}'$ with conclusion $\Gamma\cup\{A\}\vdash B$ is a subderivation of $\mathcal{D}$, and so is *smaller* in the sense of the inductive structure. Now we can assume that P holds for D', and use that to prove that the property holds for D.

$(\to E)$: (This is the second inductive case) In this case, we have that $\mathcal{D}'$ is of the structure:

$$\dfrac{\begin{array}{c}\boxed{\quad\mathcal{D}_1\quad}\\ \Gamma\vdash A\to B\end{array}\qquad\begin{array}{c}\boxed{\mathcal{D}_2}\\ \Gamma\vdash A\end{array}}{\Gamma\vdash B}$$

Now the derivations $\mathcal{D}_1$ and $\mathcal{D}_2$ are subderivations of $\mathcal{D}$, and we can assume that P holds for both, and use that to prove that P for $\mathcal{D}$.

In each case, we can use the fact that we know what is the last rule applied, which gives us not only existence of the sub-derivations, but also all other properties that are specified in the rule.

We can always add formulas to the set of assumptions without invalidating the judgement. This property is called *weakening*:

Lemma 9.31 (WEAKENING) *If $\Gamma\vdash A$ and $\Gamma\subseteq\Gamma'$, then also $\Gamma'\vdash A$.*

We leave the proof as Exercise 57.

Definition 9.32 Let the size of a formula be defined by:

$$\begin{array}{rcl} size(\top) & \triangleq & 1\\ size(\bot) & \triangleq & 1\\ size((A\to B)) & \triangleq & size(A)+size(B)+1\end{array}$$

We call a proof shaped like

$$\dfrac{\dfrac{\begin{array}{c}\boxed{\quad\mathcal{D}_1\quad}\\ \Gamma\cup\{A\}\vdash B\end{array}}{\Gamma\vdash A\to B}\,(\to I)\qquad\begin{array}{c}\boxed{\mathcal{D}_2}\\ \Gamma\vdash A\end{array}}{\Gamma\vdash B}\,(\to E)$$

a *cut*, and define the size of the cut to be the size of the cut type, which is the conclusion of the $(\to I)$ step, $A\to B$.

We can remove cuts by letting the proof for the right-hand judgement, $\mathcal{D}_2 :: \Gamma\vdash A$, take the place in the left-hand proof, $\mathcal{D}_1$, of all the applications of the axiom $\Gamma'\cup\{A\}\vdash A$, thus creating

$$\begin{array}{c}\begin{array}{c}\boxed{\mathcal{D}_2}\\ \Gamma\vdash A\end{array}\quad\cdots\quad\begin{array}{c}\boxed{\mathcal{D}_2}\\ \Gamma\vdash A\end{array}\\ \boxed{\qquad\qquad\mathcal{D}_1'\qquad\qquad}\\ \Gamma\vdash B\end{array}$$

We aim to show that for every derivable result $\Gamma\vdash A$ there exists a cut-free proof that shows $\Gamma\vdash A$, by systematically performing this operation. This is not straightforward, since removing a cut can generate a cut:

Example 9.33 Take the proof

$$\dfrac{\dfrac{\dfrac{\dfrac{}{B\to B,B\vdash B\to B}\,(Ax)\qquad\dfrac{}{B\to B,B\vdash B}\,(Ax)}{B\to B,B\vdash B}\,(\to E)}{B\vdash(B\to B)\to B\to B}\,(\to I)\qquad\dfrac{\dfrac{}{B\vdash B}\,(Ax)}{\vdash B\to B}\,(\to I)}{\vdash B\to B}\,(\to E)$$

This contains the (single) cut

$$\dfrac{\dfrac{\dfrac{\boxed{\mathcal{D}_1}}{B\to B, B\vdash B}}{B\vdash (B\to B)\to B\to B}\,(\to I) \qquad \dfrac{\boxed{\mathcal{D}_2}}{\vdash B\to B}}{\vdash B\to B}\,(\to E)$$

We create a new proof by letting $\mathcal{D}_2$ take the position of

$$\dfrac{}{B\to B, B\vdash B\to B}\,(Ax)$$

in $\mathcal{D}_1$. This results in:

$$\dfrac{\dfrac{\dfrac{}{B\vdash B}\,(Ax)}{\vdash B\to B}\,(\to I) \qquad \dfrac{}{B\vdash B}\,(Ax)}{B\vdash B}\,(\to E)$$

which again contains a cut, not present before. However, the *size* of the cut-type has decreased.

We can use this observation to show our result.

First, using weakening, we show the following:

Lemma 9.34 Let $\mathcal{D}$ be the derivation

$$\dfrac{\dfrac{\dfrac{\boxed{\mathcal{D}_1}}{\Gamma\cup\{A\}\vdash B}}{\Gamma\vdash A\to B}\,(\to I) \qquad \dfrac{\boxed{\mathcal{D}_2}}{\Gamma\vdash A}}{\Gamma\vdash B}\,(\to E)$$

Assume that $A\to B$ is the largest cut-type in $\mathcal{D}$ (in the sense that there are no other cuts with types larger than or equal to size$(A\to B)$). Then there exists a proof for $\Gamma\vdash B$ that has at most cuts with cut types smaller than size$(A\to B)$.

Proof: By induction on the structure of proofs (derivations), following the structure of $\mathcal{D}_1$.

(Ax): Then $\mathcal{D}_1$ is shaped like

$$\dfrac{}{\Gamma\cup\{B\}\vdash A}$$

and either:

$(B\equiv A)$: Then we have $\mathcal{D}_2 :: \Gamma\vdash A$, which has only cuts smaller than *size*$(A\to B)$; no new cuts get introduced, so the result holds.

$(B\not\equiv A)$: Then $A\in\Gamma$, so $\Gamma=\Gamma'\cup\{A\}$ and by rule (Ax) we also have

$$\dfrac{}{\Gamma'\cup\{A\}\vdash A}$$

a cut-free proof; so, in particular, $\Gamma\vdash A$.

$(\to I)$: Then $B=C\to D$ for some C and D, and $\mathcal{D}_1$ is shaped like

$$\dfrac{\dfrac{\boxed{\mathcal{D}_3}}{\Gamma\cup\{A\}\cup\{C\}\vdash D}}{\Gamma\cup\{A\}\vdash C\to D}\,(\to I)$$

Notice that by induction there exists a proof $\mathcal{D}_3'$ such that

$$\dfrac{\boxed{\mathcal{D}_3'}}{\Gamma\cup\{C\}\vdash D}$$

all cuts in $\mathcal{D}_3'$ are smaller than *size*$(A\to B)$. We apply $(\to I)$ to that proof and obtain a proof for

$$\dfrac{\begin{array}{c}\boxed{\mathcal{D}'_3}\\ \Gamma\cup\{C\}\vdash D\end{array}}{\Gamma\vdash C\to D}\;(\to I)$$

Notice that we have not added a cut with this step, so the property holds.

$(\to E)$: Then $\mathcal{D}_1$ is of the shape

$$\dfrac{\begin{array}{c}\boxed{\mathcal{D}_3}\\ \Gamma\cup\{A\}\vdash C\to B\end{array}\qquad\begin{array}{c}\boxed{\mathcal{D}_4}\\ \Gamma\cup\{A\}\vdash C\end{array}}{\Gamma\cup\{A\}\vdash B}\;(\to E)$$

for some C. By induction, there exist $\mathcal{D}'_3 :: \Gamma\vdash C\to B$ and $\mathcal{D}'_4 :: \Gamma\vdash C$ such that all cuts in $\mathcal{D}'_3$ and $\mathcal{D}'_4$ are smaller than $size(A\to B)$. Applying rule $(\to E)$ to these will give the proof

$$\dfrac{\begin{array}{c}\boxed{\mathcal{D}'_3}\\ \Gamma\vdash C\to B\end{array}\qquad\begin{array}{c}\boxed{\mathcal{D}'_4}\\ \Gamma\vdash C\end{array}}{\Gamma\vdash B}\;(\to E)$$

Since it is possible that $\mathcal{D}'_3$ ends with rule $(\to I)$, we might add a cut here, but with $size(B)$, which is smaller than $size(C\to B)$. ■

We can now state:

Theorem 9.35 *Let $\mathcal{D} :: \Gamma\vdash A$; then there exists a cut-free proof for $\Gamma\vdash A$.*

Proof: By double induction; the outermost goes by complete induction to the size of the largest cut type in $\mathcal{D}$, and the innermost goes by mathematical induction to the number of cuts in $\mathcal{D}$ with a cut type of that size.

Let

$$\dfrac{\dfrac{\begin{array}{c}\boxed{\mathcal{D}_1}\\ \Gamma\cup\{A\}\vdash B\end{array}}{\Gamma\vdash A\to B}\;(\to I)\qquad\begin{array}{c}\boxed{\mathcal{D}_2}\\ \Gamma\vdash A\end{array}}{\Gamma\vdash B}\;(\to E)$$

be a sub-derivation such that $A\to B$ is the largest cut type in $\mathcal{D}$, and all cut types inside $\mathcal{D}_1$ and $\mathcal{D}_2$ are smaller. By Lemma 9.34 we can remove this cut, and replace this sub-derivation with one for $\Gamma\vdash B$ which has a smaller largest cut type, thus decreasing the number of largest cuts by 1, or decreasing the size of the largest cut type in $\mathcal{D}$.

In either case, the proof follows by induction. ■

Example 9.36 The proof of Example 9.28 contains a cut. Removing it yields:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\{A\to B,B,A\}\vdash A}\;(Ax)}{\{A\to B,A\}\vdash B\to A}\;(\to I)}{\{A\to B,A\}\vdash A\to B\to A}\;(\to I)\qquad\dfrac{}{\{A\to B,A\}\vdash A}\;(Ax)}{\{A\to B,A\}\vdash B\to A}\;(\to E)\qquad\begin{array}{c}\dfrac{\dfrac{}{\{A\to B,A\}\vdash A\to B}\;(Ax)\qquad\dfrac{}{\{A\to B,A\}\vdash A}\;(Ax)}{\{A\to B,A\}\vdash B}\;(\to E)\\ \vdots\\ \vdots\end{array}}{\{A\to B,A\}\vdash A}\;(\to E)}{\{A\to B\}\vdash A\to A}\;(\to I)}{\emptyset\vdash (A\to B)\to A\to A}\;(\to I)$$

This again has a cut; removing it yields:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\{A\to B,B,A\}\vdash A}\,(Ax)}{\{A\to B,A\}\vdash B\to A}\,(\to I) \qquad \dfrac{\dfrac{}{\{A\to B,A\}\vdash A\to B}\,(Ax) \quad \dfrac{}{\{A\to B,A\}\vdash A}\,(Ax)}{\{A\to B,A\}\vdash B}\,(\to E)}{\dfrac{\{A\to B,A\}\vdash A}{\{A\to B\}\vdash A\to A}\,(\to I)}\,(\to E)}{\emptyset\vdash(A\to B)\to A\to A}\,(\to I)$$

This again has a cut; removing it yields:

$$\dfrac{\dfrac{\dfrac{}{\{A\to B,A\}\vdash A}\,(Ax)}{\{A\to B\}\vdash A\to A}\,(\to I)}{\emptyset\vdash(A\to B)\to A\to A}\,(\to I)$$

which is a cut-free proof.

There is also a notion of induction over well-founded orderings, but we will not discuss that here. An example would be a proof by induction on a lexicographical ordering, which for the countable case corresponds to double induction; this technique thereby corresponds to what we used in the proof of Theorem 9.35. Remark that, by the axiom of well ordering, there exists a well-ordering on $\mathbb{R}$, so we have in principle a way of showing properties over $\mathbb{R}$ using induction; if only we knew what that ordering is . . .

Exercises

Exercise 52 Show that, for all $n\in\mathbb{N}$: $\Sigma_{i=0}^{n}\, i^2 = (n\times(n+1)\times(2n+1))/6$.

Exercise 53 Even, the set of even numbers, is inductively defined through:

- $0\in$ *Even.*
- *if $k\in$ Even, then $k{+}2\in$ Even.*

Prove that, for every even number n, $n\times m$ is even for any natural number m.

Exercise 54 Let the set $V\subseteq\mathbb{Z}$ (so addition and subtraction are defined as usual) be defined through the grammar:

$$v \;::=\; 0 \mid (v+3) \mid (v-4)$$

Show that $\mathbb{N}\subseteq V$.

Exercise 55 Take the notion of strings, concatenation of strings, length of a string and reverse of a string as in Example 9.12. Show, for all strings s_1 and s_2, that

1) $\mathit{length}(s_1\circ s_2) = \mathit{length}(s_1)+\mathit{length}(s_2)$.

2) $\mathit{rev}(s_1\circ s_2) = \mathit{rev}(s_2)\circ\mathit{rev}(s_1)$.

Exercise 56 • *The* height *of a proof in* IL $\mathcal{D}$, *$ht(\mathcal{D})$, is inductively defined by:*

(Ax): *Then $\mathcal{D} :: \Gamma\vdash A$, where $A\in\Gamma$, and $ht(\mathcal{D})=1$.*

$(\to I)$: *Then $\mathcal{D} :: \Gamma\vdash A\to B$, and $\mathcal{D}$ has a sub-proof $\mathcal{D}' :: \Gamma\cup\{A\}\vdash B$. Then $ht(\mathcal{D}) = ht(\mathcal{D}')+1$.*

$(\to E)$: *Then $\mathcal{D} :: \Gamma\vdash B$ has two sub-proofs $\mathcal{D}_1 :: \Gamma\vdash A\to B$ and $\mathcal{D}_2 :: \Gamma\vdash A$. Then $ht(\mathcal{D}) = \max(ht(\mathcal{D}_1), ht(\mathcal{D}_2))+1$.*

• *The* complexity *of a proof $\mathcal{D}$, $comp(\mathcal{D})$, is inductively defined by (we focus on the last rule applied):*

(Ax): *Then $\mathcal{D} :: \Gamma\vdash A$, where $A\in\Gamma$, and $comp(\mathcal{D})=2$.*

$(\to I)$: *Then $\mathcal{D} :: \Gamma\vdash A\to B$, and $\mathcal{D}$ has a sub-proof $\mathcal{D}' :: \Gamma\cup\{A\}\vdash B$. Then $comp(\mathcal{D}) = comp(\mathcal{D}')+1$.*

$(\to E)$: *Then $\mathcal{D} :: \Gamma\vdash B$ has two sub-proofs $\mathcal{D}_1 :: \Gamma\vdash A\to B$ and $\mathcal{D}_2 :: \Gamma\vdash A$. Then $comp(\mathcal{D}) = comp(\mathcal{D}_1)+comp(\mathcal{D}_2)$.*

Show that, for all proof, $ht(\mathcal{D}) < comp(\mathcal{D})$.

Exercise 57 *If $\Gamma \vdash A$ and $\Gamma \subseteq \Gamma'$, then also $\Gamma' \vdash A$.*

Exercise 58 *Assume that $x \notin fv(L)$. Show that $M[N/x][L/y] = M[L/y][N[L/y]/x]$.*

Exercise 59 *Show that, when $M =_\beta N$, then there are terms $M_1, M_2, \ldots, M_n, M_{n+1}$ such that $M \equiv M+1$, $N \equiv M_{n+1}$, and, for all $1 \leq i \leq n$, either $M_i \to_\beta^* M_{i+1}$, or $M_{i+1} \to_\beta^* M_i$.*