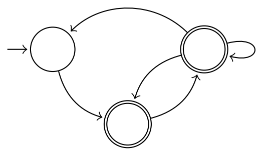


AN INTRODUCTION TO Formal Languages AND Automata Theory

Luke Collins
maths.com.mt/notes

Version 0.1[*] (November 29, 2020)

CONTENTS

[*]If you find any mathematical, grammatical or typographical errors whilst reading these notes, please let the author know via email: luke.collins@um.edu.mt.

I. INTRODUCTION

In this introductory chapter, we will briefly go over the basic mathematical prerequisites needed in order to study formal languages and automata.

LOGIC

Statements are sentences which are decidedly **true** or **false**. Examples of statements include "Today is a rainy day", "$1+1=5$" or "for all n, the number n^2+1 is not a square", whereas examples of non-statements are "What time is it?", "Edam cheese", "73" or "This statement is false".[1]

We define the following *connectives* to make new statements from old ones.

Definitions 1.1 (Logical connectives)**.** Let Φ and Ψ denote statements.

(i) $\neg\Phi$ denotes the statement "not Φ" called the *negation of* Φ, which is defined to be true precisely when Φ is false, and vice-versa.

(ii) $\Phi \wedge \Psi$ denotes the statement "Φ and Ψ", called the *conjunction of* Φ *and* Ψ, which is defined to be true precisely when Φ and Ψ are both true, and false otherwise.

(iii) $\Phi \vee \Psi$ denotes the statement "Φ or Ψ", called the *disjunction of* Φ *and* Ψ, which is defined to be false precisely when Φ and Ψ are both false, and true otherwise.

(iv) $\Phi \Rightarrow \Psi$ denotes the statement "if Φ, then Ψ" or "Φ implies Ψ", called the (*material*) *implication of* Φ *and* Ψ, which is defined to be false precisely when Φ true and Ψ is false, and true otherwise.

(v) $\Phi \Leftrightarrow \Psi$ denotes the statement "Φ if and only if Ψ" or "Φ iff Ψ" (for short), called the *bi-implication* or *biconditional of* Φ *and* Ψ, which is defined to be true precisely when Φ and Ψ are the same (i.e., both true or both false), and false otherwise.

Examples 1.2. Let Φ = "$1+1=2$", Ψ = "Pigs can fly" and Ξ = "$7<3$". These are true, false and false respectively. We have:

(i) $\neg\Phi$ is "$1+1\neq 2$", which is false, since Φ is true. $\neg\Psi$ is "Pigs cannot fly", which is true, and $\neg\Xi$ is "$7 \geqslant 3$", which is also true.

[1] This is an interesting one. Why is it not a statement? Well, statements must be either true or false; so let us suppose that it is true. We immediately see the contradiction that arises: it claims itself to be false, so it cannot be true. Suppose therefore that it is false. In this case, it makes a claim which is true — contradicting that it should be false. Thus we cannot say that the statement is true, nor that it is false.

(ii) $\Phi \wedge \Psi$ is "$1+1=2$ and pigs can fly", which is false, since Ψ is false.

(iii) $\Psi \vee \Xi$ is "Pigs can fly or $7 < 3$". This is false, since both are false. On the other hand, $\Psi \vee \Phi$ is "Pigs can fly or $1+1=2$". Since Φ is true, this is true.

(iv) $\Psi \Rightarrow \Xi$ is "If pigs can fly, then $7 < 3$". This is true since Ψ, i.e., "pigs can fly" is false. On the other hand, $\Phi \Rightarrow \Psi$ is "If $1+1=2$, then pigs can fly" is false, whereas $\Psi \Rightarrow \Psi$ which is "If $1+1=2$ then $1+1=2$" is true.

(v) $\Psi \Leftrightarrow \Xi$ is "Pigs can fly if and only if $7 < 3$" is true, because Ψ and Ξ are both false. $\Phi \Leftrightarrow \neg\Psi$ is "$1+1=2$ if and only if pigs cannot fly" is true.

Notation. We also have the following notation which we will be using throughout.

- The symbol '$:=$' denotes the *assignment operator*, i.e. we write

$$a := b$$

to mean that 'a' is defined to be 'b'. Equivalently, we write $b =: a$.

- The symbol '$==$' denotes the *comparison operator*, i.e.

$$a == b \iff a = b.$$

The difference between '$a = b$' and '$a == b$' is only superficial and the two statements are essentially telling us the same thing; however we make use of the latter to emphasise that a comparison is being made, and that the result is a boolean value (**true** or **false**).

- The symbol $\forall$ is shorthand for the phrase "for all" or "for every", and the symbol $\exists$ is shorthand for "there exists" or "for some".

For example,

$$\forall x(x > 5 \Rightarrow x > 4),$$

in words, is "for all x, if x is larger than 5, then x is larger than 4.

Another example:

$$(\forall\varepsilon > 0)(\exists\delta > 0)(0 < |x - x_0| < \delta \Rightarrow |f(x) - \ell| < \varepsilon),$$

expressed in words, becomes "for all $\varepsilon > 0$, there exists $\delta > 0$ such that if $0 < |x - x_0| < \delta$, then $|f(x) - \ell| < \varepsilon$."

SET THEORY

Next we go to the notion of a *set*. Informally, a set is a *collection of distinct "objects"*. In particular, the defining characteristic of a set is the idea of *membership*—an object x is either a member of a set S, or not. We write $x \in S$ for "*x is an element of the set S*" (or x is in S), and similarly $y \notin S$ for the negation "*y is* not *an element of S*". Sets may be defined by listing their elements between curly brackets, e.g.

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

defines the set A whose elements are $1, 2, 3, 4$ and 5. We have $1 \in A$, but $0 \notin A$ (for example). It is conventional to use capital letters for sets.

Notice that our definition of a set is an imprecise one: we do not clearly state what counts as an "object" in a set, nor which sets we are explicitly allowed construct; we simply define sets by describing their elements verbally. Can sets contain other sets? Are we allowed to define strange sets such as "the set of all sets which are not members of themselves"? This vagueness leads to fundamental problems in mathematics and philosophy—but we will not concern ourselves with these issues here.[2]

Some of the important sets which we encounter are:

- The **empty set**, denoted by the symbol $\varnothing$, is the set such that

$$x \notin \varnothing \text{ for all } x.$$

- The set of **natural numbers**, denoted by the symbol $\mathbb{N}$, is the infinite set containing all positive whole numbers and zero:

$$\mathbb{N} = \{0, 1, 2, 3, 4, \dots\}.$$

- The set of **integers**, denoted by the symbol $\mathbb{Z}$ (for the German *zählen*, meaning *counting*), is the infinite set containing the positive whole numbers, the negative whole numbers and zero:

$$\mathbb{Z} = \{\dots, -2, -1, 0, 1, 2, \dots\}.$$

[2] The interested reader is encouraged to look up the graphic novel *Logicomix* to get an idea of the historical significance of these problems, or, to get stuck in to formal set theory itself, take a look at the textbook *Introduction to Set Theory* by Hrbáček and Jech.

- The set of **rational numbers**, denoted by the symbol $\mathbb{Q}$ (for *quotient*), is the set of all numbers which can be expressed as a ratio of two integers. For example, this set contains the numbers $\frac{1}{2}$, $\frac{22}{7}$, $-\frac{1}{3}$, 0 and 5 $(= \frac{5}{1})$.
- The set of **real numbers**, denoted by the symbol $\mathbb{R}$, contains all of the rational numbers, together with all the numbers which have infinitely many digits after the decimal point. Some of these are rational (e.g. $\frac{1}{3} = 0.333\ldots$ and $\frac{1}{7} = 0.142857142\ldots$), but others are *irrational*, that is, not rational (e.g. $\sqrt{2} = 1.41421\ldots$, $\pi = 3.14159\ldots$ and $e = 2.7182818\ldots$).

  It is not easy to see that some numbers are irrational. The easiest number to prove is irrational is $\sqrt{2}$, and a proof can be seen here.

Notice that each of the sets we defined ($\varnothing$, $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$) contains all the elements of the previous one. When a set B contains all the elements of A, or more formally, if

$$\forall x\,(x \in A \implies x \in B),$$

we say A is a *subset* of B and write $A \subseteq B$. For example, if $A = \{1, 2, 3\}$, $B = \{1, 2, 3, 4\}$, then $A \subseteq B$. Note that by this definition, every set S is a subset of itself. Also note that it is not necessarily the case that for two given sets, one set is a subset of the other or vice-versa; for example if $C = \{2, 4, 6, 8\}$, we neither have $A \subseteq C$ nor $C \subseteq A$.[3]

If A contains every element of B, and B contains every element of A, that is, if both $A \subseteq B$ and $B \subseteq A$, we say that A is *equal to* B, written $A = B$. Observe that

$$\varnothing \subseteq \mathbb{N} \subseteq \mathbb{Z} \subseteq \mathbb{Q} \subseteq \mathbb{R}$$

but none of these are equal. In particular, by noting that $\sqrt{2} \in \mathbb{R}$ but $\sqrt{2} \notin \mathbb{Q}$, we see that $\mathbb{Q} \neq \mathbb{R}$.

Notation (Set Comprehension)**.** Here we introduce an alternative notation to describe sets, instead of explicitly listing their elements. Suppose we want to describe the set of even numbers, E. Since there are infinitely many, in our current notation, we are forced to use ellipses $(\ldots)$ and let the reader deduce what the set contains:

$$E = \{\ldots, -4, -2, 0, 2, 4, \ldots\}.$$

[3] Unlike the similar looking relation "$\leqslant$" for real numbers, where it *must* be the case that $x \leqslant y$ or $y \leqslant x$. Because of this, $\subseteq$ is called a *partial order*, and $\leqslant$ is called a *total order*.

There is a level of ambiguity with this notation however (for instance, one might think the members of E are plus or minus the powers of 2).

Alternatively, using set comprehension, we write this as

$$E = \{2n : n \in \mathbb{Z}\},$$

where the colon is read "*such that*". The whole expression is read as "$E =$ the set of all things of the form $2n$, such that $n \in \mathbb{Z}$"; in other words, E is the set of even numbers. In general, the notation

$$X = \{x : \Phi(x)\}$$

defines the set of all things "x" which satisfy the statement Φ. Sometimes, we would like to take our elements from a larger set. For example, the set of prime numbers can be written as

$$P = \{n \in \mathbb{N} : n \text{ is prime}\}$$

or as

$$P = \{n : n \in \mathbb{N} \text{ and } n \text{ is prime}\}.$$

In general, there are many ways to express the same set using comprehension.

Examples 1.3. We give some examples of set comprehension.

$$\begin{aligned}
\{n^2 : n \in \mathbb{N}\} &= \{1, 4, 9, 16, \dots\} \\
\{5n : n \in \mathbb{Z}\} &= \{\dots, -5, 0, 5, 10, \dots\} \\
\{x \in \mathbb{Q} : x + 2 = 1\} &= \{-1\} \\
\{x \in \mathbb{N} : x + 2 = 1\} &= \varnothing \\
\{2x : x \in \mathbb{R}\} &= \mathbb{R} \\
\{\tfrac{a}{b} : a, b \in \mathbb{Z} \text{ and } b \neq 0\} &= \mathbb{Q}
\end{aligned}$$

Now, let us define some set operations, that is, ways to combine sets to create new sets.

Definitions 1.4. Suppose A and B are two sets. Then

(i) The *union of A and B*, denoted $A \cup B$, is the set defined by the property

$$\text{If } x \in A \vee x \in B, \text{ then } x \in A \cup B.$$

(ii) The *intersection of A and B*, denoted $A \cap B$, is the set defined by the property

$$\text{If } x \in A \wedge x \in B, \text{ then } x \in A \cap B.$$

(iii) The *difference between A and B*, denoted $A \smallsetminus B$, is the set defined by the property

$$\text{If } x \in A \wedge x \notin B, \text{ then } x \in A \smallsetminus B.$$

Examples 1.5. If $A = \{1, 2, 3, 4, 5\}$, $B = \{2, 4, 6, 8, 10\}$ and $C = \{-1, 0, 1\}$, then

$$\begin{aligned}
A \cup B &= \{1, 2, 3, 4, 5, 6, 8, 10\} \\
A \cap B &= \{2, 4\} \\
A \smallsetminus B &= \{1, 3, 5\} \\
(A \cup B) \cap C &= \{1, 2, 3, 4, 5, 6, 8, 10\} \cap \{-1, 0, 1\} = \{1\} \\
A \cup (B \cap C) &= \{1, 2, 3, 4, 5\} \cup \varnothing = \{1, 2, 3, 4, 5\} = A
\end{aligned}$$

We also have the notion of a power set.

Definition 1.6. Given a set A, the *power set of A* is the set

$$\{A : A \subseteq X\}$$

of all subsets of X and is denoted by $\wp X$.

Example 1.7. For example,

$$\wp\{1, 2, 3\} = \{\varnothing, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\}$$

and

$$\begin{aligned}
\wp\wp\{1, 2\} = \big\{&\varnothing, \{\varnothing\}, \{\{1\}\}, \{\{2\}\}, \{\{1, 2\}\}, \{\varnothing, \{1\}\}, \{\varnothing, \{2\}\}, \\
&\{\varnothing, \{1, 2\}\}, \{\{1\}, \{2\}\}, \{\{1\}, \{1, 2\}\}, \{\{2\}, \{1, 2\}\}, \\
&\{\varnothing, \{1\}, \{2\}\}, \{\varnothing, \{1\}, \{1, 2\}\}, \{\varnothing, \{2\}, \{1, 2\}\}, \\
&\{\{1\}, \{2\}, \{1, 2\}\}, \{\varnothing, \{1\}, \{2\}, \{1, 2\}\}\big\}
\end{aligned}$$

Notice that $B \subseteq A$ if and only if $B \in \wp A$, and $A \nsubseteq \wp A \nsubseteq \wp\wp A$, etc.

FUNCTIONS AND TUPLES

In formal mathematics, we try to encode everything using sets—and when we say everything, we mean *everything*. As noted in the last section, $\mathbb{N}$ denotes the set of natural numbers and zero, i.e.,

$$\mathbb{N} = \{0, 1, 2, 3, 4, \ldots\}.$$

But what are numbers exactly? Can we think of them as being "constructed" from a more basic kind of object? The answer is yes, and in mathematics, we choose sets to be our building blocks for everything else.

Formally, we define the set $\mathbb{N}$ as the *closure* of the set $\{\varnothing\}$ under the successor operation $x \mapsto x \cup \{x\}$. This means that $\mathbb{N}$ is the smallest set containing $\varnothing$, together with any other set(s) which can be obtained by repeatedly applying the operation $x \mapsto x \cup \{x\}$. The successor of each natural number represents the next natural number in the usual order, so identifying 0 with the empty set, we have

$$\begin{aligned}
&0 = \varnothing \\
&1 = 0 \cup \{0\} = \varnothing \cup \{0\} = \{0\} = \{\varnothing\} \\
&2 = 1 \cup \{1\} = \{0, 1\} = \{\varnothing, \{\varnothing\}\} \\
&3 = 2 \cup \{2\} = \{0, 1, 2\} = \{\varnothing, \{\varnothing\}, \{\varnothing, \{\varnothing\}\}\} \\
&4 = 3 \cup \{3\} = \{0, 1, 2, 3\},
\end{aligned}$$

and so on. In general, each natural number k ends up being structurally represented as the set of all its predecessors:

$$k = \{0, 1, 2, \ldots, k - 1\}.$$

There is nothing inherently special about this way of encoding $\mathbb{N}$. This is just one of many ways to construct a set which captures the behaviour of $\mathbb{N}$, namely, a set where we have

- a least element 0, (i.e., $\varnothing$), and
- a way to always "add 1" (i.e., the successor operation $x \mapsto x \cup \{x\}$).

When encoded this way, the natural numbers are called von Neumann ordinals. The two properties above capture entirely the essence of the natural numbers, and by constructing an object which behaves this way, we showed that it is possible to encode $\mathbb{N}$ with sets alone, and we don't need to think of numbers

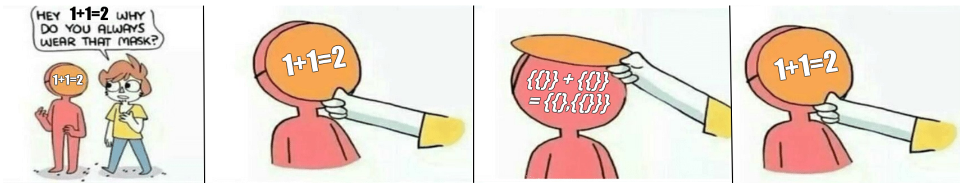


Figure 1: Set theoretic "meme"

as autonomous "objects" in their own right.[4] One can define a notion of "addition" and "multiplication" and so on, all in terms of the underlying set representations, and prove things like $1+1=2$ (refer to figure 1). But of course in practice, we still think of the natural numbers as usual, only now safer in the knowledge that they are built on a more fundamental idea which we understand well. Similarly, we can construct $\mathbb{Z}$, $\mathbb{Q}$ and $\mathbb{R}$ in terms of sets, but we will not get into that here.

Now we start to explore the idea of a function in a formal way. We need the following idea first.

Definition 1.8 (Input-output pair)**.** Given two elements $a \in A$ and $b \in B$ from the two sets A and B, the set $\{\{a\},\{a,b\}\} \in \wp\wp(A \cup B)$ is called an *input-output pair*, which we will denote by $\langle\!\langle a, b\rangle\!\rangle$. In this case, we call a the *input* and b the *output* of the pair, respectively.

It is a simple exercise in the definition of set equality to verify that input-output pairs have the property that $\langle\!\langle a, b\rangle\!\rangle = \langle\!\langle c, d\rangle\!\rangle$ if and only if $a = c$ and $b = d$, and moreover, unlike sets, input-output pairs satisfy $\langle\!\langle a, b\rangle\!\rangle \neq \langle\!\langle b, a\rangle\!\rangle$ unless $a = b$.

The set of all input-output pairs with inputs in the set A and outputs in the set B will be denoted by $A \divideontimes B$, that is,

$$A \divideontimes B := \{x \in \wp\wp(A \cup B) : (\exists a \in A)(\exists b \in B)(x = \langle\!\langle a, b\rangle\!\rangle)\}.$$

Definition 1.9 (Relation)**.** Let A, B be two sets. A (*binary*) *relation* from A to B is any subset R of $A \divideontimes B$. If $\langle\!\langle a, b\rangle\!\rangle \in R$, we say a and b are related by R and write $a \mathrel{R} b$.

Sometimes we use special symbols instead of letters for relations. For example, $\leqslant$ is a relation from $\mathbb{R}$ to $\mathbb{R}$, and $x \leqslant y \iff \langle\!\langle x, y\rangle\!\rangle \in {\leqslant}$, where ${\leqslant} \subseteq \mathbb{R} \divideontimes \mathbb{R}$.

[4]This is analogous to how any photo, video or text on a computer is just 1s and 0s at the lowest level, but we still don't think about them in terms of 1's and 0's!

Example 1.10. Consider the sets $A = \{1, \ldots, 6\}$ and $B = \{\mathit{odd}, \mathit{even}, \mathit{prime}\}$. We can define the relation $\sim \subseteq A \divideontimes B$ which relates the numbers in A to the words in B which describe them. For example $1 \sim \mathit{odd}$ and $2 \sim \mathit{prime}$, but $4 \nsim \mathit{prime}$ (that is, $\langle\!\langle 4, \mathit{prime} \rangle\!\rangle \notin \sim$). Visually, we can think of the relation as a subset of all possible 'arrows' from elements in the set A to those of the set B (as in figure 2).

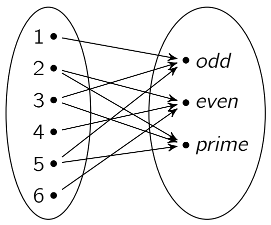


Figure 2: The relation $\sim \subseteq A \divideontimes B$

Definition 1.11 (Domain and codomain)**.** If R is a relation from the set A to the set B, then the set A is called the *domain* of R, which we denote by $\mathrm{dom}(R)$, and the set B is called the *codomain* of R, which we denote by $\mathrm{cod}(R)$.

A function is basically a special kind of relation.

Definition 1.12 (Function)**.** A *function* is a relation from a set A to a set B such that

$$(\forall a \in A)\left(\left((\exists b \in B)(\exists b' \in B)(\langle\!\langle a, b \rangle\!\rangle \in f \wedge \langle\!\langle a, b' \rangle\!\rangle \in f)\right) \implies b = b'\right).$$

In other words, a function from A to B is a set of input-output pairs such that for any element $a \in A$ in the domain, there is at most one input-output pair $\langle\!\langle a, b \rangle\!\rangle \in f$. (Indeed, the condition is saying that if $\langle\!\langle a, b \rangle\!\rangle$ and $\langle\!\langle a, b' \rangle\!\rangle$ are both in f, then b and b' must be the same, so that "two" input-output pairs are actually the same one.)

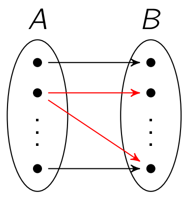


Figure 3: A function cannot relate an element in A to more than one in B

If for $a \in A$ there exists an input-output pair $\langle\!\langle a, b\rangle\!\rangle \in f$, then we write $f(a) = b$. Moreover, we interpret the notation $f(a)$ alone to stand for the unique $b \in B$ for which $\langle\!\langle a, b\rangle\!\rangle \in f$.

For example, if we have the sets $A = \{1, 2, 3, 4\}$, $B = \{10, 20, 30\}$ and then define $f = \{\langle\!\langle 1, 20\rangle\!\rangle, \langle\!\langle 2, 20\rangle\!\rangle, \langle\!\langle 4, 10\rangle\!\rangle\}$, then f is a function from A to B. Moreover, $f(1) = 20$, and $f(4)$ is the value 10.

Remark 1.13. The notation $f(a)$ only makes sense for functions. Indeed, the relation $\sim$ from example 1.10 is not functional, since some elements in A are related to more than one $b \in B$ (and therefore not *at most* one as in definition 1.12). So for example, since both $2 \sim$ *even* and $2 \sim$ *prime*, we cannot make sense of $f(2)$, since it could either be referring to *even* or *prime* (where f is representing $\sim$).

Another example: recall that a square root of a number $x \in \mathbb{R}$ is some number $y \in \mathbb{R}$ such that $y^2 = x$. Recall also that for $x > 0$, there are always two possible square roots of x, one positive, the other negative. By convention, we use the symbol $\sqrt{x}$ to denote the positive root of x. If we do not assume such a convention, and write for example, $\sqrt{4} = \pm 2$ instead, then if we write $\sqrt{4}$ alone, it is not clear which of $+2$ or -2 this notation is referring to.

To show that a function f has domain A and codomain B, we sometimes write f in "full" as $f\colon A \rightharpoonup B$ (where the arrow is indicative of the phrase "from A to B"). A function $f\colon A \rightharpoonup B$ is said to be *total* if for all $a \in A$, there exists some $b \in B$ such that $f(a) = b$. In other words, a function f is total if every

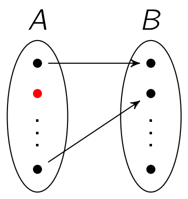


Figure 4: A total function cannot leave an element in A unrelated to any element in B

a in the domain has a pair in f (unlike in the example we just gave, where $3 \in A$ had no input-output pair in f, see figure 4). To show that f is a total function with domain A and codomain B, we denote it by writing $f\colon A \to B$ instead of $f\colon A \rightharpoonup B$. If a function is not total, i.e., if there is at least one $a \in A$ with no existing input-output pair in f, then we say f is *partial*.

Definition 1.14 (Domain restriction)**.** Let $f\colon A \rightharpoonup B$ be a function, and let $A' \subseteq A$. Then we define the function $(f \restriction A')\colon A' \to B$, read f *restricted to* A', by

$$f \restriction A' := \{\langle\!\langle a, b\rangle\!\rangle \in f : a \in A'\},$$

or equivalently,

$$(f \restriction A')(a) := f(a)$$

for $a \in A'$.

For example, if $f\colon \{1,2,3,4\} \rightharpoonup \{10,20,30\}$ is the example we gave earlier, then $f \restriction \{1,2\} = \{\langle\!\langle 1,20\rangle\!\rangle, \langle\!\langle 2,20\rangle\!\rangle\}$, and $f \restriction \{1,2,4\} = f$, but $f \restriction \{1,2,4\}$ is total, unlike f, which is partial.

Definition 1.15 (Domain of definition)**.** Suppose $f : A \rightharpoonup B$ is a function. The largest subset $A' \subseteq A$ such that for all $a' \in A'$, there exists $b \in B$ such that $f(a') = b$, is called the *domain of definition* of f, denoted $\operatorname{ddf}(f)$.

Equivalently, $\operatorname{ddf}(f)$ is the largest $A' \subseteq A$ such that $f \restriction A'$ is total.

For the example, $\operatorname{ddf}(f) = \{1,2,4\}$ since $3 \in A$ had no corresponding input-output pair in f.

If a function f is total, then $\operatorname{ddf}(f) = \operatorname{dom}(f)$.

Similarly, we have a name for the largest subset $B' \subseteq B$ such that for all $b' \in B'$, there exists $a \in A$ such that $f(a) = b'$. This set is called the *range* of the function f, denoted $\operatorname{ran}(f)$. For the example, $\operatorname{ran}(f) = \{10,20\}$, since $30 \in B$ had no input-output pair in f.

Thus given a function $f\colon A \rightharpoonup B$, then $\operatorname{ddf}(f)$ is the subset of A which is "used" by f, whereas $\operatorname{ran}(f)$ is the subset of B which is "used" by f.

Now we give names to some special kinds of functions.

Definitions 1.16 (Injection, surjection, bijection)**.** Suppose $f\colon A \rightharpoonup B$ is a function. Then

(i) the function f is said to be *an injection*, *injective* or *one-to-one* if

$$(\forall a_1 \in A)(\forall a_2 \in A)\big(f(a_1) = f(a_2) \implies a_1 = a_2\big).$$

Equivalently, a function is injective if every element b in the codomain has *at most one* element $a \in A$ such that $f(a) = b$.

Observe that the example f we gave earlier was not an injection since it has $f(1) = f(2)$ but $1 \neq 2$.

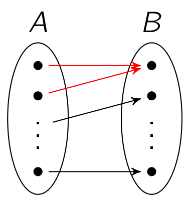


Figure 5: An injection cannot relate two elements in A to the same element in B

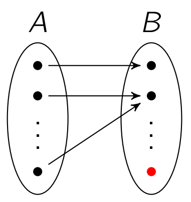


Figure 6: An surjection cannot leave an element in B unrelated to any element in A

(ii) the function f is said to be *a surjection*, *surjective* or *onto* if

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

In other words, no element of the codomain is 'unused' by the function (and therefore $\operatorname{ran}(f) = \operatorname{cod}(f) = B$).

Again, the example we gave was not surjective since $30 \in B$ was unused by f.

(iii) the function f is said to be *a bijection*, *bijective* or *a one-to-one correspondence* if it is total, injective and surjective.

An example of a bijection is $g \colon A \to B$ where $A = \{1, 2, 3\}$, $B = \{10, 20, 30\}$, and $g(a) = 10a$ for all $a \in A$.

Definition 1.17 (Cardinality)**.** For a finite set X, the *cardinality* of X, denoted $\#X$, is the number $k \in \mathbb{N}$ of elements contained in the set X.

More formally, we say that $\#A = k \in \mathbb{N}$ if and only if there exists a bijection between the set A and $k = \{0, \dots, k-1\}$. (All this is doing is basically labelling the items in the set A as the 0th, 1st, ..., $(k-1)$st, effectively counting how many there are.) This is the precise definition because it can be generalised to infinite sets, but we will not discuss it here.[5]

Definition 1.18 (k-tuple)**.** Given a finite set A, a *k-tuple* (where $k \geqslant 0$), or simply a *tuple*, is a total function $\boldsymbol{t} \colon k \to A$.

Effectively, a k-tuple encodes a sequence of k elements from A which in this context we call *entries*, ordered by the function. We will denote tuples using bold letters such as $\boldsymbol{s}$ or $\boldsymbol{\sigma}$, to distinguish them from other sets or functions.

[5] Also, we haven't really defined what it is to be "finite" in a formal way, and this is basically how we do it; we say A is finite if there exists a natural number k such that we can put it in bijection with A. Otherwise A is infinite.

A tuple is typically written by listing its entries in order and enclosing them in parentheses:

$$(a_1, a_2, \dots, a_k).$$

So for example, (c, a, b) denotes the 3-tuple $\boldsymbol{t}\colon 3 \to \{a, b, c\}$ such that $\boldsymbol{t}(0) = c$, $\boldsymbol{t}(1) = a$, and $\boldsymbol{t}(2) = b$.

Observe that two tuples $(a_1, a_2, \dots, a_k)$ and $(b_1, b_2, \dots, b_\ell)$ are equal if and only if $k = \ell$ and $a_i = b_i$ for all $i = 1, 2, \dots, k$. This follows by definition of set equality.

Definition 1.19 (Cartesian product)**.** Consider any two sets A and B. The *Cartesian product* $A \times B$ is the set

$$\{(a, b) : a \in A \wedge b \in B\}$$

of pairs (2-tuples) having the first entry in A and the second entry in B. In general, for any k sets $A_1, A_2, \dots, A_k$, the Cartesian product $A_1 \times A_2 \times \cdots \times A_k$ is the set

$$\{(a_1, a_2, \dots, a_k) : a_1 \in A_1 \wedge a_2 \in A_2 \wedge \cdots \wedge a_k \in A_k\}$$

of all k-tuples having the first entry in A_1, the second entry in A_2, …, and the kth entry in A_k.

For example, if $A_1 = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$, $A_2 = \{\alpha, \beta\}$ and $A_3 = \{\heartsuit, \diamondsuit\}$, then

$$\begin{aligned} A_1 \times A_2 \times A_3 = \{&(\mathtt{a}, \alpha, \heartsuit), (\mathtt{a}, \alpha, \diamondsuit), (\mathtt{a}, \beta, \heartsuit), (\mathtt{a}, \beta, \diamondsuit), \\ &(\mathtt{b}, \alpha, \heartsuit), (\mathtt{b}, \alpha, \diamondsuit), (\mathtt{b}, \beta, \heartsuit), (\mathtt{b}, \beta, \diamondsuit), \\ &(\mathtt{c}, \alpha, \heartsuit), (\mathtt{c}, \alpha, \diamondsuit), (\mathtt{c}, \beta, \heartsuit), (\mathtt{c}, \beta, \diamondsuit)\}. \end{aligned}$$

If $A_1 = A_2 = \cdots = A_k =: A$, then $A_1 \times A_2 \times \cdots \times A_k$ is also denoted by A^k. For example, $\{\mathtt{a}, \mathtt{b}\}^4 = \{(\mathtt{a}, \mathtt{a}, \mathtt{a}, \mathtt{a}), (\mathtt{a}, \mathtt{a}, \mathtt{a}, \mathtt{b}), (\mathtt{a}, \mathtt{a}, \mathtt{b}, \mathtt{a}), \dots, (\mathtt{b}, \mathtt{b}, \mathtt{b}, \mathtt{b})\}$.

Remark 1.20. A subtle consequence of this definition is that $A^1 \neq A$. In fact,

$$A^1 = \{(a) : a \in A\}.$$

So if $A = \{1, 2, 3\}$, then $A^1 = \{(1), (2), (3)\}$, the set of tuples of length 1.[6] This might seem like a pedantic distinction; however later on it plays an important role in differentiating between symbols and strings.

We also have $A^0 = \{()\}$ for any A, where $()$ is the empty tuple, i.e., the bijection $\epsilon\colon 0 \to A$ which is equal to the empty set (since it contains no input-output pairs).

[6]The tuple (a) is different from a since (a) denotes the bijection $\{\langle 0, x\rangle\}$.

INDUCTION

Induction is a technique which allows us to prove statements about structures which can be ordered in some way. Let us formalise the idea of "order".

Definition 1.21 (Partial order)**.** Let A be a set. A *partial order* on A is a relation $\preccurlyeq$ satisfying the following properties for all $a, b, c \in A$.

i. $a \preccurlyeq a$ (reflexivity)
ii. If $a \preccurlyeq b$ and $b \preccurlyeq a$, then $a = b$, (antisymmetry)
iii. If $a \preccurlyeq b$ and $b \preccurlyeq c$, then $a \preccurlyeq c$, (transitivity)

The pair $(A, \preccurlyeq)$ together are referred to as a *partially ordered set*, or a *poset*.

Given two elements x, y in a poset, we say they are *comparable* if either $x \preccurlyeq y$ or $y \preccurlyeq x$. Otherwise we say they are *incomparable*. A poset where any two elements are comparable is called a (*totally*) *ordered set* or a *chain*.

Let us give a few examples of posets.

Examples 1.22. (i) The real numbers $\mathbb{R}$ with the usual order $\leqslant$. Similarly $\mathbb{N}, \mathbb{Z}, \mathbb{Q}$ are all posets under $\leqslant$. (They are actually chains.)

(ii) Given any set A, the power set $\wp A$ is a poset under the order $\subseteq$. This is not a chain, since some subsets may not be comparable, e.g., $\{1, 2\}$ and $\{2, 3\}$ as subsets of $\{1, 2, 3\}$.

(iii) The set $\mathbb{N}$ with the "divides" relation is a poset (e.g., $3 \mid 12$). This is not a chain either, since $2 \nmid 3$ and $3 \nmid 2$.

(iv) The set E of words in the English language is totally ordered by the usual dictionary order.

(v) The members of a family form a poset, ordered by generation.

We can visualise posets set using a *Hasse diagram*. For instance, the poset $(\wp\{a, b, c, d\}, \subseteq)$ can be seen in figure 7. Similarly, the set of all divisors of 24 and the divides relation can be seen in figure 8. A Hasse diagram for the members of a family would just be the family tree.

Let us now focus on chains, i.e., posets in which any two elements are comparable. Even if a set A is totally ordered, it is not enough to allow us to determine the "next" element following a given $x \in A$ under the order (this will play an important role in induction). What would the "next" number following 1 in $\mathbb{R}$ under $\leqslant$?

$$\begin{tikzcd}[column sep=tiny]
 & & & & & \{a,b,c,d\} & & & & & \\
 & & \{a,b,c\} \arrow[urrr] & & \{a,b,d\} \arrow[ur] & & \{a,c,d\} \arrow[ul] & & \{b,c,d\} \arrow[ulll] & & \\
\{a,b\} \arrow[urr] \arrow[urrrr] & & \{a,c\} \arrow[u] \arrow[urrrr] & & \{a,d\} \arrow[u] \arrow[urr] & & \{b,c\} \arrow[ullll] \arrow[urr] & & \{b,d\} \arrow[ullll] \arrow[u] & & \{c,d\} \arrow[ullll] \arrow[ull] \\
 & & \{a\} \arrow[ull] \arrow[u] \arrow[urr] & & \{b\} \arrow[ullll] \arrow[urr] \arrow[urrrr] & & \{c\} \arrow[ullll] \arrow[u] \arrow[urrrr] & & \{d\} \arrow[ullll] \arrow[u] \arrow[urr] & & \\
 & & & & & \varnothing \arrow[ulll] \arrow[ul] \arrow[ur] \arrow[urrr] & & & & &
\end{tikzcd}$$

Figure 7: Hasse diagram for the poset $(\wp\{a, b, c, d\}, \subseteq)$.

$$\begin{tikzcd}
 & 24 & & & \\
8 \arrow[ur] & & 12 \arrow[ul] & & \\
 & 4 \arrow[ul] \arrow[ur] & & 6 \arrow[ul] & \\
 & & 2 \arrow[ul] \arrow[ur] & & 3 \arrow[ul] \\
 & & & 1 \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

Figure 8: Hasse diagram for the poset $(\{1, 2, 3, 4, 6, 8, 12, 24\}, \mid)$.

Definition 1.23 (Minimum)**.** Let $(A, \preccurlyeq)$ be a chain, and let $B \subseteq A$ be a non-empty subset of A. Then $m \in B$ is a *minimum* of B under $\preccurlyeq$, written

$$m = \min_{\preccurlyeq} B$$

or simply $m = \min B$, if $m \preccurlyeq b$ for every $b \in B$.

This is quite an intuitive definition, e.g., 1 is a minimum of the set $\{1, 2, 3, 4\}$ seen as a subset of $\mathbb{N}$ (or $\mathbb{R}$) with the usual order $\leqslant$. Alternatively, the English word "a" is the minimum of the entire English language in dictionary order.

We analogously have the notion of a maximum. We say that m is a maximum of $B \subseteq A$, written $m = \max B$, if $b \preccurlyeq m$ for all $b \in B$.

Theorem 1.24. *Let $(A, \preccurlyeq)$ be a chain. If a subset $B \subseteq A$ has a minimum, then the minimum is unique.*

Proof. Suppose B has two minima, say b and b'. By definition, we have $b \preccurlyeq b'$ and $b' \preccurlyeq b$. But then by antisymmetry, $b = b'$. Hence any two minima are equal. □

Okay, so minima are unique. But do they always exist? The answer is no:

Example 1.25. The set

$$S = \{x \in \mathbb{R} : 0 < x < 1\}$$

has no minimum under the usual order $\leqslant$ for $\mathbb{R}$. Indeed, suppose $m \in S$ is the minimum. Then $0 < \frac{m}{2} < 1$ so $\frac{m}{2} \in S$ also, and $\frac{m}{2} \leqslant m$ and $\frac{m}{2} \neq m$, so this contradicts that m is the minimum.

Indeed, an even simpler counterexample is $\mathbb{R}$ itself, being a subset of $\mathbb{R}$, but also having no minimum. It is this point which brings us to the stronger notion of a well-order:

Definition 1.26. Let A be a set, and let $\preccurlyeq$ be a total ordering on A. We say $\preccurlyeq$ is a *well-order* on A if every non-empty subset $B \subseteq A$ has a least element; i.e., $\min_{\preccurlyeq} B$ exists for all non-empty $B \subseteq A$.

The pair $(A, \preccurlyeq)$ together are referred to as a *well-ordered chain*.

If a set can be well-ordered, then we automatically obtain the "next" element according to the order. Indeed, we have:

Definition 1.27 (Successor)**.** Let $(A, \preccurlyeq)$ be a well-ordered chain, and let $a \in A$. Then the *successor* of a, denoted $a^+_\preccurlyeq$ or simply a^+, is defined as

$$a^+ := \min(A \setminus \{x \in A : x \preccurlyeq a\}).$$

This is quite an intuitive definition. For example, to find the successor of 5 in $\mathbb{N}$ (with the usual $\leqslant$ order), we first form the set $\{n \in \mathbb{N} : n \leqslant 5\} = \{0, 1, 2, 3, 4, 5\}$, and remove it from $\mathbb{N}$ to get $\mathbb{N} \setminus \{0, 1, 2, 3, 4, 5\} = \{6, 7, \dots\}$. The minimum of this set is 6, so $5^+ = 6$. Similarly, if you grab a dictionary and wish to find the successor of the last word on page 30, simply rip out pages 1–30. The first word on page 31 will be the successor.

Example 1.28. Not all elements of a well-ordered set are successors. Clearly the minimum element of the set is not the successor of any other element, but there can be other elements which aren't successors either.

For instance, take the set

$$A = \{1 - \tfrac{1}{n} : n \in \mathbb{N}\} \cup \{1\} = \{0, \tfrac{1}{2}, \tfrac{2}{3}, \tfrac{3}{4}, \tfrac{4}{5}, \dots, 1\}$$

with the usual order of real numbers. This defines a well-order on the set, but repeatedly applying the successor operation to the least element 0 will never give 1, i.e., the number 1 is not the successor of some other element under this order.

We will not be able to do induction on sets which are ordered this way.[7] Instead, we will focus our attention on sets where every element is a successor:

Definition 1.29 (Finitely inducible)**.** Let $(A, \preccurlyeq)$ be a well-ordered chain. We say this chain is *finitely inducible* if all elements apart from $\min A$ are successors.

Example 1.30. The set of natural numbers $\mathbb{N}$ under the usual order is finitely inducible.

In a finitely inducible chain, we can define the idea of a predecessor:

Definition 1.31 (Predecessor)**.** Let $(A, \preccurlyeq)$ be a well-ordered chain, and let $a \in A$ such that $a \neq \min A$. Then the *predecessor* of a, denoted $a^-_\preccurlyeq$ or simply a^-, is the element $a^- \in A$ such that $(a^-)^+ = a$.

We can show that analogously to the successor, the predecessor can be written as a maximum:

[7] There is an analogue to induction for sets like these called transfinite induction, but we will not discuss it in these notes.

Proposition 1.32. *Let $(A, \preccurlyeq)$ be a well-ordered chain, and let $a \in A$ such that $a \neq \min A$. Then $a^- = \max(A \smallsetminus \{x \in A : a \preccurlyeq x\})$.*

Example 1.33. In $\mathbb{N}$ under the usual order, the predecessor of 5 is

$$\max(\mathbb{N} \smallsetminus \{x \in \mathbb{N} : 5 \leqslant x\}) = \max(\mathbb{N} \smallsetminus \{5, 6, \dots\}) = \max(\{1, 2, 3, 4\}) = 4.$$

Finally we can state the main theorem of this section:

Theorem 1.34 (Principle of (finite) induction)**.** *Let $(A, \preccurlyeq)$ be a finitely inducible well-ordered chain, let $m = \min A$, and let $\Phi(a)$ be a statement about some $a \in A$. If*

(i) $\Phi(m)$ is true, and

(ii) for all $k \in A$, if $\Phi(k)$ is true, then $\Phi(k^+)$ is true,

then $\Phi(a)$ is true for all $a \in A$.

Proof. Suppose this is not the case, i.e., $\Phi(a)$ is false for some $a \in A$. Then the set $A_0 = \{a \in A : \Phi(a) \text{ is false}\}$ is non-empty, and we can determine $a_0 = \min A_0$.

By (i), $a_0 \neq m$, and so we can obtain a_0^-. Clearly $\Phi(a_0^-)$ is true since otherwise a_0 would not be $\min A'$. But then applying (ii) with $k = a_0^-$ we get that $\Phi((a_0^-)^+) = \Phi(a_0)$ is true, which is a contradiction. □

If we rephrase theorem 1.34 specifically for $(\mathbb{N}, \leqslant)$, we get the following.

Theorem 1.35 (Induction for $\mathbb{N}$)**.** *Let $\Phi(n)$ be a statement about natural numbers. Then if*

(i) $\Phi(0)$ is true, and

(ii) for all $k \in \mathbb{N}$, if $\Phi(k)$ is true, then $\Phi(k+1)$ is true,

then $\Phi(n)$ is true for all $n \in \mathbb{N}$.

What this theorem is telling us is that if we wish to prove something about natural numbers, we can do so by proving the following:

(i) it is true for $n = 0$,

(ii) if it is true for $n = k$, it is true for $n = k + 1$.

This is the so-called *domino effect*. Think of the natural numbers as an infinite line of dominoes. If the first domino falls ($n = 0$), and we know that if any kth domino falls, the one in front of it (the k^+th one) will fall, what can we conclude? That all dominoes fall!

If it is not immediately clear, combining (i) and (ii) gives us that the 0^+th domino will fall, i.e. that domino 1 will fall. But now we can use this fact and (ii) again to conclude that domino 1^+ = domino 2 will fall. We can now use this to conclude that domino 2^+ = domino 3 will fall, and so on.

Establishing $\Phi(0)$ is called the *base case*. Proving that $\Phi(k) \Rightarrow \Phi(k^+)$ for any k is called the *inductive step*, and the assumption $\Phi(k)$ here is called the *inductive hypothesis* (IH).

Example 1.36. Let us prove a very popular formula by induction. The sum of the first n positive integers:

$$1 + 2 + \cdots + n = \tfrac{n}{2}(n+1).$$

Let the formula above be the statement $\Phi(n)$. Clear $\Phi(0)$ is true, because the left-hand side is $\sum_{k=1}^{0} k = 0 = \frac{0}{2}(0+1)$, which equals the right-hand side. So the base case is done.

Next we assume $\Phi(k)$ is true, i.e. that $1 + 2 + \cdots + k = \frac{k}{2}(k+1)$. This is the inductive hypothesis.

Now can we prove $\Phi(k+1)$? We have:

$$\begin{aligned} 1 + 2 + \cdots + k + (k+1) &= \tfrac{k}{2}(k+1) + (k+1) && \text{(by the IH)} \\ &= (k+1)(\tfrac{k}{2} + 1) \\ &= \tfrac{k+1}{2}(k+2) \\ &= \tfrac{k+1}{2}((k+1)+1), \end{aligned}$$

in other words, we have that $\Phi(k+1)$ is true! This concludes the proof. $\square$

Make sure you understand this proof, in that it conforms with the requirements of theorem 1.35. Understand *why* we are allowed to use $\Phi(k)$ in the inductive step.

Example 1.37. We show that the number $4^n + 2$ is always divisible by 3.

We let $\Phi(n) = (\exists \alpha \in \mathbb{N})(4^n + 2 = 3\alpha)$, because this is what it means to be divisible by 3.

For $\Phi(0)$, we have $4^0 + 2 = 1 + 2 = 3 = 3(1)$, so we take $\alpha = 1$ and this completes the base case.

Now assume $\Phi(k)$ holds, i.e. $4^k + 2 = 3\alpha$ for some $\alpha \in \mathbb{N}$. Can we show $\Phi(k+1)$?

$$\begin{aligned}
4^{k+1} + 2 &= 4(4^k) + 2 \\
&= 4(4^k) + 4 - 2 \\
&= 4(4^k + 1) - 2 \\
&= 4(4^k + 2 - 1) - 2 \\
&= 4(3\alpha - 1) - 2 \qquad \text{(by IH)} \\
&= 12\alpha - 4 - 2 \\
&= 12\alpha - 6 \\
&= 3(4\alpha - 2) \\
&= 3\beta \qquad \text{where } \beta = (4\alpha - 2) \in \mathbb{N}
\end{aligned}$$

i.e. $\Phi(k+1)$ is true. By theorem 1.35, we have that $\Phi(n)$ holds for all $n \in \mathbb{N}$.

Example 1.38. How about an example with matrices. We prove that if

$$\mathbf{A} = \begin{pmatrix} 3 & 1 \\ 0 & 2 \end{pmatrix},$$

then the matrix power $\mathbf{A}^n$ is given by

$$\mathbf{A}^n = \begin{pmatrix} 3^n & 3^n - 2^n \\ 0 & 2^n \end{pmatrix}.$$

For the base case, we have

$$\mathbf{A}^0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 3^0 & 3^0 - 2^0 \\ 0 & 2^0 \end{pmatrix},$$

so the statement holds when $n = 0$. Now

$$\begin{aligned}
\mathbf{A}^{k+1} &= \mathbf{A}^k\mathbf{A} \\
&= \begin{pmatrix} 3^k & 3^k - 2^k \\ 0 & 2^k \end{pmatrix}\begin{pmatrix} 3 & 1 \\ 0 & 2 \end{pmatrix} \qquad \text{(by the IH)} \\
&= \begin{pmatrix} 3^k(3) + 0 & 3^k + 2(3^k - 2^k) \\ 0 & 2^k(2) \end{pmatrix}
\end{aligned}$$

$$= \begin{pmatrix} 3^{k+1} & 3^k + 2(3^k) - 2^{k+1} \\ 0 & 2^{k+1} \end{pmatrix}$$

$$= \begin{pmatrix} 3^{k+1} & 3^k(1+2) - 2^{k+1} \\ 0 & 2^{k+1} \end{pmatrix} = \begin{pmatrix} 3^{k+1} & 3^{k+1} - 2^{k+1} \\ 0 & 2^{k+1} \end{pmatrix},$$

as required. □

Notice that in the last example, we do not explicitly define a predicate $\Phi(n)$, nor do we actually state the inductive hypothesis. This is typically how one presents an inductive argument, since the predicate is always simply a copy of the result we wish to prove, as is the inductive hypothesis (simply with k instead of n).

The general strategy of proof by induction is to try and "break down" the objects we are working with in the $n = k + 1$ case into the $n = k$ case, plus something else which elevates it to the $n = k + 1$ case (e.g. the matrix $\mathbf{A}^{k+1}$ into $\mathbf{A}^k\mathbf{A}^1$). This allows one to apply the inductive hypothesis.

Thus, one deduces that induction would not be very useful when it is very difficult, or impossible, to relate the case of $n = k$ to that of $n = k+1$. When results are about the natural numbers themselves, often one can use induction, because we have an explicit way to relate k and $k + 1$ (namely the successor k^+). Let us give one final example with series, which emphasises this strategy of breaking the case $n = k + 1$ down into the $n = k$ case "plus other stuff".

Example 1.39. We prove that

$$\sum_{r=n}^{2n} r(r+2) = \frac{n}{6}(n+1)(14n+19).$$

For the base case, we have

$$\sum_{r=0}^{0} r(r+2) = 0 = \frac{0}{6}(0+1)(14(0)+19),$$

so the statement holds.[8]

[8]Sometimes, in cases where the $n = 0$ feels a bit vacuous, we prove the $n = 1$ case instead. Technically, if one does this and excludes the zero case, this proves the result for $n \geqslant 1$, and not for all $n \in \mathbb{N}$. But it's fine usually, because the reason we consider $n = 1$ in the first place is because $n = 0$ is not interesting. In this case, $n = 1$ gives $\sum_{r=1}^{2} r(r+2) = 1(1+2) + 2(2+2) = 11 = \frac{1}{6} \cdot 2 \cdot 33 = \frac{1}{6}(1+1)(14+19)$, so the case $n = 1$ holds. At least it feels like we've actually proved something!

Now when $n = k + 1$,

$$\begin{aligned}
&\sum_{r=k+1}^{2(k+1)} r(r+2) \\
&= \underbrace{\sum_{r=k}^{2k} r(r+2)}_{\text{the } n=k \text{ case}} \underbrace{- \sum_{r=k}^{k} r(r+2) + \sum_{r=2k+1}^{2k+2} r(r+2)}_{\text{“other stuff”}} \\
&= \frac{k}{6}(k+1)(14k+19) - k(k+2) + (2k+1)(2k+3) + (2k+2)(2k+4) \\
&\hphantom{=} \qquad \text{(by IH)} \\
&= \frac{1}{6}(14k^3 + 75k^2 + 127k + 66) \quad ^{9} \\
&= \frac{k+1}{6}(k+2)(14k+33) = \frac{k+1}{6}((k+1)+1)(14(k+1)+19),
\end{aligned}$$

as required. □

We close this section with a theorem known as “strong induction”, which allows us to strengthen the induction hypothesis by assuming $\Phi(t)$ for all $t \preccurlyeq k$, instead of just $\Phi(k)$.

Theorem 1.40 (Principle of strong (finite) induction)**.** *Let $(A, \preccurlyeq)$ be a finitely inducible well-ordered chain, let $m = \min A$, and let $\Phi(a)$ be a statement about some $a \in A$. If*

(i) $\Phi(m)$ is true, and

(ii) for all $k \in A$, if $\Phi(t)$ is true for every $t \preccurlyeq k$, then $\Phi(k^+)$ is true,

then $\Phi(a)$ is true for all $a \in A$.

Proof. Let $\Psi(a) =$ “$\Phi(r)$ is true for all $r \preccurlyeq a$”. Notice that this implies $\Phi(a)$, so if we prove that (i) and (ii) imply $\Psi(a)$ for all $a \in A$, we would be done. We proceed by normal induction (theorem 1.34).

$\Psi(m)$ is equivalent to $\Phi(m)$, which is true by (i), this completes the base case. Now the inductive hypothesis $\Psi(k)$ allows us to deduce $\Phi(k^+)$ by applying (ii). Thus we have that $\Psi(k^+)$ is true. This completes the proof. □

[9] Even though this cubic might seem like a headache to factorise (factor theorem, etc.), remember that *we know* what this should equal if the result is true; it should be the right-hand side of the result with $k+1$ in place of n, i.e. $\frac{k+1}{6}((k+1)+1)(14(k+1)+19)$—so we can use this fact to make an “educated guess” about the factors.

RECURSION

A very important result, which goes hand-in-hand with induction, is the idea of definition by recursion. This idea is essential in mathematics and computer science, the idea of a function "using itself" in a definition.

Let us motivate this with an example, before we state the theorem. The factorial of a natural number n, written $n!$, is the product of all numbers from 1 to n, i.e.,

$$n! := n \cdot (n-1) \cdots 2 \cdot 1.$$

An alternative way to define this is the following: suppose we want to find 5!, *but we already know what* 4! *is*. How can we use this information? Well if we already know what 4! is, we can just multiply this by 5 and we get 5!—in general, we have that $n! = n \cdot (n-1)!$. We could use this to define factorial differently—let $\mathrm{fac}(n)$ denote our "newly" defined factorial.

If we define $\mathrm{fac}(n) = n \cdot \mathrm{fac}(n-1)$, this almost works. Indeed, if we try to evaluate 5! this way, we do

$$\begin{aligned}
\mathrm{fac}(5) &= 5 \cdot \mathrm{fac}(4) \\
&= 5 \cdot 4 \cdot \mathrm{fac}(3) \\
&= 5 \cdot 4 \cdot 3 \cdot \mathrm{fac}(2) \\
&= 5 \cdot 4 \cdot 3 \cdot 2 \cdot \mathrm{fac}(1) \\
&= 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 \cdot \mathrm{fac}(0) \qquad (*) \\
&= 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 \cdot 0 \cdot \mathrm{fac}(-1) \\
&\vdots
\end{aligned}$$

Something seems to have gone wrong here. The "unrolling" of the fac's seems to have worked, because we did get the product $5 \cdots 1$ appearing, but this will keep on going forever. We need some mechanism which stops this infinite expansion. If we instead say $\mathrm{fac}(0) = 1$, then we would actually stop at the line $(*)$. This is called the *base case*. Familiar?

So we define

$$\mathrm{fac}(n) := \begin{cases} 1 & \text{if } n = 0 \\ n \cdot \mathrm{fac}(n-1) & \text{otherwise.} \end{cases}$$

And this works! Apart from being nicely succinct (without having to use informal notions such as $\cdots$), this is how one would program a factorial function on a computer. Indeed, in Python, one would write

```
def fac(n):
    if n==0: return 1
    else: return n * fac(n-1)
```

to define the factorial of n. Or, if you are used to C-like languages, you would do

```
unsigned fac(unsigned n){
    return (n == 0) ? 1 : n * fac(n-1);
}
```

But is this always allowed? Can we define a random function, say,

$$f(n) = \begin{cases} 1 & \text{if } n = 0 \\ nf(n+1) & \text{otherwise?} \end{cases}$$

Try to work out $f(5)$. It's not hard to see why this definition is problematic. Thus, we need to see precisely *when* we are allowed to do this. Indeed, notice that fac(n) made use of n, and of *the value of* fac($n-1$). In other words, in defining $f(n)$, if we only allow $f(n-1)$ to appear (and possibly n), then the definition should be fine.

This is what the following theorem guarantees.

Theorem 1.41 (Definition by Recursion)**.** *Suppose A is a finitely inducible well-ordered chain, let X be any set, let $x \in X$ and let $g\colon A \times X \to X$ be a total function. Then there exists a unique total function $f\colon A \to X$ such that*

$$f(a) = \begin{cases} x & \textit{if } a = \min A \\ g(a, f(a^-)) & \textit{otherwise.} \end{cases}$$

All this theorem is telling us is that we are allowed to have $f(a^-)$ appearing in the definition of f (and only that), and this will determine a unique, valid function from A to X. We will not prove it here, since it is a bit technical (although not difficult). You probably guessed it, but the proof is by induction!

For $n!$, the function $g\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ we need to take $g(a, t) = at$.

Example 1.42. Let $A = \{1 - \frac{1}{n} : n \in \mathbb{N}\} = \{0, \frac{1}{2}, \frac{2}{3}, \frac{3}{4}, \dots\}$, this is a finitely inducible well-ordered chain under the usual order $\preccurlyeq$. Define $f(0) = 0$ and $f(a) = f(a^-) + a$. We see that $f(a)$ is the sum $0 + \frac{1}{2} + \frac{2}{3} + \cdots + a$.

In this case, $g\colon A \times \mathbb{R} \to \mathbb{R}$ is $g(a, t) = t + a$.

RECURSION AND INDUCTION ON POSETS

In this section, we will see that we can generalise the recursion and induction theorems to posets. The notation $m \prec b$ is an abbreviation for $m \preccurlyeq b$ and $m \neq b$, which we call the *strict order* of $\preccurlyeq$.

In a poset A, unlike in a chain, if we have a minimum, it is not always unique. We denote by $M(A)$ the set of minima of the poset:

Definition 1.43 (Minimal set)**.** Let $(A, \preccurlyeq)$ be a poset. The *minimal set* of a subset $B \subseteq A$, denoted $M_{\preccurlyeq}(B)$ or just $M(B)$, is the set defined by

$$M(B) \coloneqq \{m \in B : \text{there is no } b \in B \text{ such that } b \prec m\}.$$

If A is a chain, then $M(A) = \{\min A\}$ (if $\min A$ exists). Indeed, just as with chains, the set $M(B)$ can sometimes be empty, e.g., if we take the set

$$A = \{\tfrac{1}{n} : n \in \mathbb{N}\},$$

we notice that with the usual order $\leqslant$, we get $M(A) = \varnothing$.

Definition 1.44 (Well-founded order)**.** A partial order $\preccurlyeq$ on A is *well-founded* if $M_{\preccurlyeq}(B) \neq \varnothing$ for any non-empty $B \subseteq A$.

This idea is the poset analogue of a well-order. Indeed, if A is totally ordered by $\preccurlyeq$, and $\preccurlyeq$ is well-founded, then $\preccurlyeq$ is well-ordered. Notice that being well-founded essentially means we do not have any infinitely descending chains like

$$\cdots \prec a_3 \prec a_2 \prec a_1.$$

Indeed, any infinite descending chain in a poset would give $M(\{a_1, a_2, \dots\}) = \varnothing$. (Proof: if $a_k \in M(\{a_1, a_2, \dots\})$, then there is no $a_{k+1} \prec a_k$, so the chain would not be infinite.) Conversely, if a poset A is not well-founded, then there is a non-empty subset $B \subseteq A$ such that $M(B) = \varnothing$. Given any $b \in B$ there must be $m_1 \in B$ such that $m_1 \prec b$ (since $b \notin M(B)$). But also there is $m_2 \prec m_1$ (since $m_1 \notin M(B)$). Similarly there is $m_3 \prec m_2$ (since $m_2 \notin M(B)$), and so on, giving rise to an infinite descending chain $\cdots \prec m_2 \prec m_1 \prec b$.

In any well-founded poset, we can do recursion. We first need to define the notion of an *initial segment*.

Definition 1.45 (Initial segment)**.** Let $(A, \preccurlyeq)$ be a well-founded poset, and pick $a \in A$. The *initial segment* of a, denoted $I[a]$, is the set of all points strictly smaller than a under the partial order, i.e.,

$$I[a] \coloneqq \{t \in A : t \prec a\}.$$

Notation. A notation which we could have introduced earlier is A^B, which denotes the set of all total functions from B to A. We also have a more general notation for unions. We write $\bigcup_{x\in X} A_x$ for the set

$$\{a : (\exists x \in X)(a \in A_a)\},$$

i.e., the set of all a's such that a is in some A_x for some $x \in X$.

The reason we need these two pieces of notation is because we will be making use of the set $\bigcup_{a\in A} X^{I[a]}$ in the statement of the next theorem. This is the set of all total functions from some initial segment $I[a]$ to a set X.

Theorem 1.46 (Recursion theorem)**.** *Suppose A is a set partially ordered by the well-founded partial order $\preccurlyeq$, let X be any set, and let $g\colon A \times S \to X$ be any total function, where $S \coloneqq \bigcup_{a\in A} X^{I[a]}$. Then there exists a unique total function $f\colon A \to X$ such that*

$$f(a) = g(a, f \restriction I[a])$$

for all $a \in A$.

In other words, f is allowed to appear in the definition of $f(a)$, as long as we restrict its inputs to points in $I[a]$, i.e., to those points $t \in A$ which are strictly less than a under the partial order.

Just as with theorem 1.41, we will not prove this here, because the proof is technical. The interested reader can find a proof in any standard textbook on set theory.

Before we illustrate this theorem with some examples, we will restate this theorem (less generally) in terms of something called the ranked partition of a poset.

Definition 1.47 (Ranked poset)**.** A poset $(A, \preccurlyeq)$ is said to be *ranked* if there are no chains of infinite length between two elements, i.e., if $a \prec b$, we can only ever have finitely many a_i's between them:

$$a = a_1 \prec a_2 \prec a_2 \prec \cdots \prec a_n \prec b.$$

Example 1.48. Recall example 1.28, where we had the set

$$A = \{1 - \tfrac{1}{n} : n \in \mathbb{N}\} \cup \{1\} = \{0, \tfrac{1}{2}, \tfrac{2}{3}, \tfrac{3}{4}, \tfrac{4}{5}, \dots, 1\}.$$

This is *not* a ranked poset, since $0 < 1$, and there is a chain of infinite length between them, i.e.,

$$0 < \tfrac{1}{2} < \tfrac{2}{3} < \cdots < 1.$$

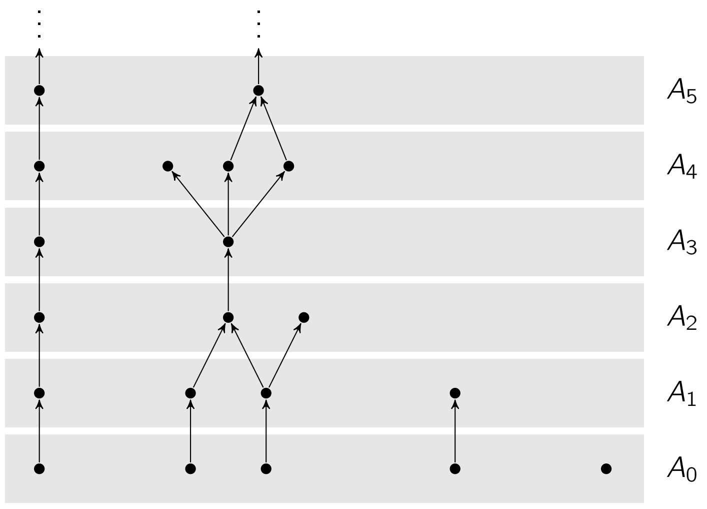


Figure 9: The Hasse diagram of a poset A and its ranked partition

This turns out to be the poset analogue of being finitely inducible. Indeed, if $\preccurlyeq$ is a total order which is ranked, then it is finitely inducible. If a poset is ranked and well-founded, we can partition its elements as follows.

Definition 1.49 (Ranked partition)**.** Let A be a set ordered by a ranked well-founded partial order $\preccurlyeq$. Define $A_n\colon \mathbb{N} \to \wp A$ by recursion as

$$A_n := M\Big(A \smallsetminus \bigcup_{k<n} A_k\Big).$$

Then the *ranked partition of* A is the family of sets $\mathcal{A} = \{A_0, A_1, \dots\}$.

Example 1.50. Consider the poset illustrated in the Hasse diagram in figure 9. As we can see, the ranked partition simply distinguishes between the different "storeys" of the Hasse diagram. Indeed, we have

$$A_0 = M\Big(A \smallsetminus \bigcup_{k<0} A_k\Big) = M(A \smallsetminus \varnothing) = M(A)$$

(since there is no $k < 0$ in $\mathbb{N}$), so the fist set in the partition is simply the minimal set of A. Next,

$$A_1 = M\Big(A \smallsetminus \bigcup_{k<1} A_k\Big) = M(A \smallsetminus A_0),$$

which is the minimal set of A once we remove the bottom layer, i.e., the second "storey". Similarly, $A_2 = M(A \smallsetminus (A_0 \cup A_1))$, and so on.

This actually is a 'partition' of the set A, i.e., a collection of non-overlapping sets which together make up the set A. In other words, we have:

Theorem 1.51. *Let A be a set, let $\preccurlyeq$ be a ranked well-founded order on A, and let $\mathcal{A} = \{A_0, A_1, \dots\}$ be its ranked partition. Then*

(i) $A_i \cap A_j = \varnothing$ for all $i \neq j$, and

(ii) $A = A_0 \cup A_1 \cup \cdots = \bigcup_{k\in\mathbb{N}} A_k$.

Proof. For (i), suppose $i < j$. Notice that for any $B \subseteq A$, by definition, if $x \in M(B)$, we must have $x \in B$. Now if $x \in A_j$, we must have

$$\begin{aligned} & x \in M\Big(A \setminus \bigcup_{k<j} A_k\Big) \\ \implies\ & x \in A \setminus \bigcup_{k<j} A_k \\ \implies\ & x \in A \qquad \text{and} \qquad x \notin \bigcup_{k<j} A_k. \\ \implies\ & x \in A \qquad \text{and} \qquad x \notin A_k \quad \text{for any } k < j, \end{aligned}$$

and in particular we get that $x \notin A_i$. Thus $A_i \cap A_j = \varnothing$.

Now for (ii), we're saying that eventually each $a \in A$ appears in some A_i. Suppose (for contradiction) that $S = \{a \in A : a \notin A_i \text{ for any } i\}$ is non-empty, and let $m \in M(S)$ (by well-foundedness). Then there exists some $t \in A$ such that $t \prec m$, and $t \in A_i$ for some i (otherwise $m \notin M(S)$). Let

$$t = t_i \prec t_{i+1} \prec \cdots \prec t_j = m$$

be a maximal chain joining t to m, i.e., there is no $s \in A$ which can be inserted between t_k and t_{k+1} (i.e., $t_k \prec s \prec t_{k+1}$) for any $i \leqslant k < j$. Such a chain exists, since we can start from $t \prec m$, and insert elements t_k between them until we reach a point where we can no longer do so (we are guaranteed to stop since A is ranked). Now let $T = \{t = t_i, t_{i+1}, \dots, t_j = m\}$, this is a well-ordered, finitely inducible chain under $\preccurlyeq$. Using induction, we can show that $t_k \in A_k$ for each $t_k \in T$, and in particular, we get that $m \in A_j$, which contradicts that $m \in S$. $\square$

We can use the idea of the ranked partition to restate the recursion theorem.

Theorem 1.52 (Recursion theorem for ranked posets)**.** *Let $(A, \preccurlyeq)$ be a ranked well-founded poset with ranked partition $\mathcal{A} = \{A_0, A_1, \dots\}$, let X be any set, let $x\colon A_0 \to X$ be a total function, and let $g\colon A\times S \to X$ be any total function, where $S := \bigcup_{k\in\mathbb{N}} X^{A_k}$. Then there exists a unique total function $f\colon A \to X$ such that*

$$f(a) = \begin{cases} x(a) & \text{if } a \in A_0 \\ g\big(a, f \restriction \bigcup_{k<n} A_k\big) & \text{if } a \in A_n,\ n > 0 \end{cases}$$

for all $a \in A$.

This looks more like theorem 1.41. Indeed, notice that if we let $A = \mathbb{N}$ with the usual order, then $A_0 = \{0\}$, and $\bigcup_{k<n} A_k = \{0\} \cup \{1\} \cup \cdots \{n^-\} = n$. Also, if g is defined by a fixed expression in a for all a, then the only term that can appear in g is $f(a^-)$, since this equality must hold for all a, and in particular, when $a \in A_1 = \{1\}$, we can only have $f(0) = f(a-1)$ appearing. Thus in the case where g is defined by the same expression for all a, we must have $g(a, f \restriction \{a^-\})$.

We can also obtain a version of theorem 1.41 with two base cases. If we order $\mathbb{N}$ such that 0 and 1 are not comparable, i.e., as in figure 10, the theorem

$$\begin{tikzcd}
1 \arrow[dr] & & & & & \\
& 2 \arrow[r] & 3 \arrow[r] & 4 \arrow[r] & 5 \arrow[r] & \cdots \\
0 \arrow[ur] & & & & &
\end{tikzcd}$$

Figure 10: Partial order on $\mathbb{N}$ for two base cases coming from $\{0, 1\} \in A_0$

becomes

$$f(n) = \begin{cases} x(n) & \text{if } n \in A_0 = \{0, 1\} \\ g\big(n, f \restriction \bigcup_{k<r} A_k\big) & \text{if } a \in A_r,\ r > 0, \end{cases}$$

and if we let $x(0) = x_0$, $x(1) = x_1$, and assume that g is defined by some fixed expression in terms of n, then in the case that $n = 2$, only $f(n-1)$ and $f(n-2)$ can make an appearance, which means that we must have

$$f(n) = \begin{cases} x_0 & \text{if } n = 0 \\ x_1 & \text{if } n = 1 \\ g(n, f \restriction \{n-1, n-2\}) & \text{otherwise.} \end{cases}$$

Example 1.53 (Fibonacci numbers). The Fibonacci sequence is defined by $f(0) = 0$, $f(1) = 1$, and $f(n) = f(n-1) + f(n-2)$ for $n \geqslant 2$, i.e., each term is the sum of the previous two. This defines a valid function since it agrees with the form above. The first few values of f are given below.

n	0	1	2	3	4	5	6	7	8	9	10
$f(n)$	0	1	1	2	3	5	8	13	21	34	55

Example 1.54 (Binomial coefficients). Another famous example is the number $\binom{n}{k}$ of k-subsets of $n = \{0, \ldots, n-1\}$, i.e.,

$$\binom{n}{k} := \#\{A \in \wp n : \#A = k\}.$$

For example, $\binom{5}{3} = 10$ since there are 10 subsets of $5 = \{0, 1, 2, 3, 4\}$ with size 3, i.e.,

$$\#\left\{\begin{matrix}\{0,1,2\}, \{0,1,3\}, \{0,1,4\}, \{0,2,3\}, \{0,2,4\}, \\ \{0,3,4\}, \{1,2,3\}, \{1,2,4\}, \{1,3,4\}, \{2,3,4\}\end{matrix}\right\} = 10.$$

It can be shown that $\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$ when $n \geqslant 1$. Indeed, let $C(n, k)$ denote the set $\{A \in \wp n : \#A = k\}$ which defines $\binom{n}{k}$, so that $\binom{n}{k} = \#C(n, k)$. Notice that

$$\begin{aligned} C(n,k) &= \{S \in C(n,k) : k^- \in S\} \cup \{S \in C(n,k) : k^- \notin S\} \\ &= \{S \cup \{k^-\} : S \in C(n-1, k-1)\} \cup C(n-1, k), \end{aligned}$$

and since for disjoint finite sets (i.e., sets with $A \cap B = \varnothing$), we have that $\#(A \cup B) = \#A + \#B$, we get that

$$\begin{aligned} \#C(n,k) &= \#\{S \cup \{k^-\} : S \in C(n-1, k-1)\} + \#C(n-1, k) \\ &= \binom{n-1}{k-1} + \binom{n-1}{k}. \end{aligned}$$

When arranged in a triangular array, we get the famous "Pascal's triangle", illustrated in figures 11 and 12. What the formula $\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$ tells us is that each entry in this triangular array is the sum of the two above it.

We can choose to define these numbers in a recursive way instead, taking this formula as inspiration. The domain of such a function, let's call it c, should be pairs (n, k) such that $k \leqslant n$, i.e., we let

$$A = \{(n, k) \in \mathbb{N}^2 : k \leqslant n\},$$

$$\begin{array}{ccccccccccccc}
 & & & & & & \binom{0}{0} & & & & & & \\
 & & & & & \binom{1}{0} & & \binom{1}{1} & & & & & \\
 & & & & \binom{2}{0} & & \binom{2}{1} & & \binom{2}{2} & & & & \\
 & & & \binom{3}{0} & & \binom{3}{1} & & \binom{3}{2} & & \binom{3}{3} & & & \\
 & & \binom{4}{0} & & \binom{4}{1} & & \binom{4}{2} & & \binom{4}{3} & & \binom{4}{4} & & \\
 & \binom{5}{0} & & \binom{5}{1} & & \binom{5}{2} & & \binom{5}{3} & & \binom{5}{4} & & \binom{5}{5} & \\
\iddots & & & & & & \vdots & & & & & & \ddots
\end{array}$$

Figure 11: Pascal's triangle, in terms of the numbers $\binom{n}{k}$

$$\begin{array}{ccccccccccccc}
 & & & & & & 1 & & & & & & \\
 & & & & & 1 & & 1 & & & & & \\
 & & & & 1 & & 2 & & 1 & & & & \\
 & & & 1 & & 3 & & 3 & & 1 & & & \\
 & & 1 & & 4 & & 6 & & 4 & & 1 & & \\
 & 1 & & 5 & & 10 & & 10 & & 5 & & 1 & \\
\iddots & & & & & & \vdots & & & & & & \ddots
\end{array}$$

Figure 12: Pascal's triangle, with the numbers evaluated

and we will have $c\colon A \to \mathbb{N}$. A possible base case for c is to take the stream of 1's (i.e. the "sides" of the triangle), so that the inside can be filled in with the "sum of the two above" formula. This corresponds to the definition

$$c(n,k) = \begin{cases} 1 & \text{if } k = 0 \text{ or } n = k \\ c(n-1,k-1) + c(n-1,k) & \text{otherwise.} \end{cases}$$

We must therefore define the partial order on A by the following properties:

- $(n,0) \preccurlyeq a$ and $(n,n) \preccurlyeq a$ for all $n \in \mathbb{N}$ and $a \in A$
- $(n-1,k) \prec (n,\ell)$ for any $n,k,\ell \in \mathbb{N}$ (where $k < n$ and $\ell \leqslant n$).

This way, our ranked partition is that illustrated in figure 13. Of course, one should verify that this is a well-founded, ranked partial order. To make our

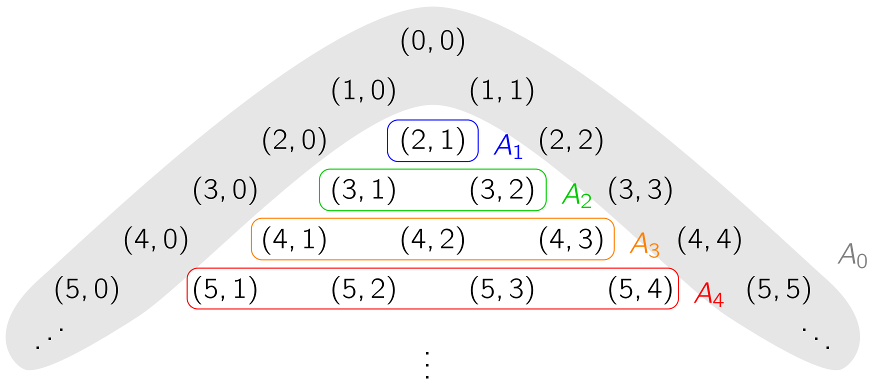

(0, 0)
(1, 0) (1, 1)
(2, 0) (2, 1) A_1 (2, 2)
(3, 0) (3, 1) (3, 2) A_2 (3, 3)
(4, 0) (4, 1) (4, 2) (4, 3) A_3 (4, 4)
(5, 0) (5, 1) (5, 2) (5, 3) (5, 4) A_4 (5, 5)
A_0

Figure 13: Ranked partition for $c(n,k)$

definition look more like theorem 1.46, we have

$$c(n,k)=\begin{cases} 1 & \text{if } (n,k)\in A_0 \\ g(n, c\restriction\bigcup_{t<r}A_t) & \text{if } (n,k)\in A_r,\ r>0, \end{cases}$$

where it is clear that because of the order, $(n-1,k-1),(n-1,k)\in\bigcup_{t<r}A_t$ for all (n,k), so our function definition is valid.

Next we discuss induction on posets. We saw in the section on induction that there are some strong requirements in order for us to prove something by induction; we saw that being a chain was not even enough, we needed a well-order which was finitely inducible. How can we hope to do induction on a poset? Well as we have seen, we have introduced analogues of these ideas for posets:

Chain		Poset
well-ordered	$\longleftrightarrow$	well-founded
$\min A$	$\longleftrightarrow$	$M(A)$
finitely inducible	$\longleftrightarrow$	ranked

Using the ranked partition of a poset, we can create a chain on which we can do induction.

Theorem 1.55. *Let A be a set, let $\preccurlyeq$ be a ranked, well-founded order on A, and let $\mathcal{A}=\{A_0,A_1,\dots\}$ be its ranked partition. Then $(\mathcal{A},\ll)$ is a well-ordered finitely inducible chain, where $\ll$ is defined by $A_i\ll A_j\Leftrightarrow i\leqslant j$.*

Proof. This follows immediately by applying the fact that $(\mathbb{N}, \leqslant)$ is well-ordered finitely inducible chain to the subscripts of the A_i. Indeed, this is a chain because we simply need to compare the subscripts of any two A_i, A_j, thus any two are comparable. Similarly, it is well-ordered because any subset $\{A_{i_1}, A_{i_2}, \dots\}$ of $\mathcal{A}$ has minimal element $A_{\min\{i_1, i_2, \dots\}}$, and finally it is finitely inducible because each A_i is the successor of A_{i^-} apart from $\min(\mathcal{A}) = A_0$. □

This allows us to state a version of the induction theorem for posets.

Theorem 1.56 (Induction on posets)**.** *Let A be a set, let $\preccurlyeq$ be a a ranked, well-founded order on A, let $\mathcal{A} = \{A_0, A_1, \dots\}$ be its ranked partition, and let $\Phi(a)$ be a statement about some $a \in A$. If*

(i) $\Phi(a)$ is true for all $a \in A_0$, and

(ii) for all $A_k \in \mathcal{A}$, if $\Phi(a)$ is true for all $a \in A_k$, then $\Phi(a)$ is true for all $a \in A_{k+1}$,

then $\Phi(a)$ is true for all $a \in A$.

Proof. This is very straightforward, simply define

$$\Psi(k) = \text{“}\Phi(a) \text{ is true for all } a \in A_k\text{”},$$

for all $k \in \mathbb{N}$, and apply theorem 1.35. □

Similarly, the “strong” analogue (theorem 1.40) is

Theorem 1.57 (Strong induction on posets)**.** *Let A be a set, let $\preccurlyeq$ be a ranked, well-founded order on A, let $\mathcal{A} = \{A_0, A_1, \dots\}$ be its ranked partition, and let $\Phi(a)$ be a statement about some $a \in A$. If*

(i) $\Phi(a)$ is true for all $a \in A_0$, and

(ii) for all $A_k \in \mathcal{A}$, if $\Phi(a)$ is true for all $a \in \bigcup_{t<k} A_t$, then $\Phi(a)$ is true for all $a \in A_{k+1}$,

then $\Phi(a)$ is true for all $a \in A$.

The proof is identical to that of theorem 1.56, but it simply invokes theorem 1.40 rather than theorem 1.34 with the same $\Psi(k)$.

Remark 1.58. The essential idea behind what was discussed in this section is the following. Any ranked well-founded poset is equivalent to a well-ordered finitely inducible chain if we consider elements on the “same storey” of the

Hasse diagram equivalent by grouping them together as one element (i.e., elements in the same minimal set).

Indeed, as we have seen, any well-founded poset gives us a well-ordered finitely inducible chain, but conversely, given any partition $\{A_1, A_2, \dots\}$ of a set A which is a well-ordered finitely inducible chain (by $\ll$, say), we can simply convert it to a ranked well-founded poset by defining the partial order

$$a \preccurlyeq b \qquad \iff \qquad a \in A_i, \quad b \in A_j \quad \text{and} \quad i \leqslant j.$$

Thus there is a very clear back-and-forth between the two.

LANGUAGES AND STRINGS

Now we define the objects which are the main focus of this set of notes: languages and strings.

Definition 1.59 (Alphabet)**.** An *alphabet* is a non-empty finite set, whose elements we call *symbols*.

By 'symbols' here we mean that the members of an alphabet are typically thought of as representing letters, characters or digits; essentially things which we are used to juxtaposing. For example, a common alphabet is the binary alphabet $\{\mathtt{0}, \mathtt{1}\}$, or the English alphabet $\{\mathtt{a}, \mathtt{b}, \mathtt{c}, \dots, \mathtt{z}\}$. We usually use the letter Σ to denote an alphabet.

Definition 1.60 (String)**.** A *string* drawn from an alphabet Σ is a tuple whose entries are members of Σ.

For example, $(\mathtt{1}, \mathtt{0}, \mathtt{1}, \mathtt{1}, \mathtt{0}, \mathtt{1})$ and $(\mathtt{1}, \mathtt{1}, \mathtt{1})$ are strings drawn from $\{\mathtt{0}, \mathtt{1}\}$, and $(\mathtt{s}, \mathtt{t}, \mathtt{e}, \mathtt{f}, \mathtt{a}, \mathtt{n}, \mathtt{i}, \mathtt{a})$ is a string drawn from the alphabet $\{\mathtt{a}, \mathtt{b}, \mathtt{c}, \dots, \mathtt{z}\}$. For convenience, we will stop using tuple notation and simply juxtapose the symbols. So instead of $(\mathtt{1}, \mathtt{0}, \mathtt{1}, \mathtt{1}, \mathtt{0}, \mathtt{1})$ we write $\mathtt{101101}$, and similarly we write $\mathtt{stefania}$ instead of $(\mathtt{s}, \mathtt{t}, \mathtt{e}, \mathtt{f}, \mathtt{a}, \mathtt{n}, \mathtt{i}, \mathtt{a})$. We will denote strings using bold letters such as $\boldsymbol{s}$ or $\boldsymbol{\sigma}$ to distinguish them from members of the alphabet.

Definition 1.61 (The Empty String)**.** The *empty string* is the tuple of length zero, and is denoted by ϵ.

This definition might cause some discomfort from a set theoretic point of view, but remember how we encoded tuples as functions; ϵ here is a bijection equal to the empty set.

Definition 1.62 (Kleene-closure)**.** Let Σ be an alphabet. The *Kleene-closure* of Σ, denoted Σ^*, is the set of strings given by

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

where $\Sigma^0 = \{\epsilon\}$.

Clearly, Σ^* is the set of all strings drawn from Σ.

Definition 1.63 (Language)**.** A *language* over the alphabet Σ is a subset L of Σ^*. We sometimes denote L by $L(\Sigma)$ to show that the strings in L have symbols from Σ.

If we ignore capitalisation and other orthographic subtleties (hyphens, and so on) then English language is a subset of Σ^* where $\Sigma = \{\mathtt{a}, \mathtt{b}, \dots, \mathtt{z}\}$. The language

$$\{\mathtt{a}, \mathtt{aa}, \mathtt{aaa}, \mathtt{aaaa}, \mathtt{aaaaa}, \dots\}$$

is a subset of Σ^*, where $\Sigma = \{\mathtt{a}, \mathtt{b}\}$. Note that this language, unlike the English language, contains infinitely many strings.

Now we define some operations on strings.

Definition 1.64 (Basic Operations)**.** Let L be a language defined on the alphabet Σ.

(i) For $\alpha \in \Sigma$ and $\boldsymbol{s} = (s_1, s_2, \dots, s_k) \in L$, we define the *concatenation of* α *and* $\boldsymbol{s}$, denoted $\alpha \,.\, \boldsymbol{s}$, by

$$\alpha \,.\, \boldsymbol{s} := (\alpha, s_1, s_2, \dots, s_k).$$

If $\boldsymbol{s} = \epsilon$, then we define $\alpha \,.\, \epsilon := (\alpha)$.[10]

(ii) For $\boldsymbol{s} = (s_1, s_2, \dots, s_k), \boldsymbol{t} = (t_1, t_2, \dots, t_\ell) \in L$, we define the *concatenation of* $\boldsymbol{s}$ *and* $\boldsymbol{t}$, denoted $\boldsymbol{s} +\!\!+ \boldsymbol{t}$, by

$$\boldsymbol{s} +\!\!+ \boldsymbol{t} := (s_1, s_2, \dots, s_k, t_1, t_2, \dots, t_\ell).$$

We also define $\boldsymbol{s} +\!\!+ \epsilon := \boldsymbol{s} =: \epsilon +\!\!+ \boldsymbol{s}$.[11]

[10] More formally, since $\boldsymbol{s} \colon k \to \Sigma$ for some k, we can give the definition of $\alpha \,.\, \boldsymbol{s} \colon k^+ \to \Sigma$ as

$$(\alpha \,.\, \boldsymbol{s})(n) := \begin{cases} \alpha & \text{if } n = 0 \\ \boldsymbol{s}(n^-) & \text{otherwise.} \end{cases}$$

[11] A more formal analogue for $(\boldsymbol{s} +\!\!+ \boldsymbol{t}) \colon (k + \ell) \to \Sigma$ where $\boldsymbol{s} \colon k \to \Sigma$ and $\boldsymbol{t} \colon \ell \to \Sigma$ is

$$(\boldsymbol{s} +\!\!+ \boldsymbol{t})(n) := \begin{cases} \boldsymbol{s}(n) & \text{if } n < k \\ \boldsymbol{t}(n - k) & \text{otherwise.} \end{cases}$$

(iii) For $\boldsymbol{s} \in L$ and $n \in \mathbb{N}$, we define the *string exponential*, denoted $\boldsymbol{s}^n$, by recursion as

$$\boldsymbol{s}^n := \begin{cases} \epsilon & \text{if } n = 0 \\ \boldsymbol{s} \mathbin{+\!\!+} \boldsymbol{s}^{n-1} & \text{otherwise.} \end{cases}$$

(iv) For $\boldsymbol{s} \in L$, we define the *length of* $\boldsymbol{s}$, denoted $|\boldsymbol{s}|$, recursively by

$$|\boldsymbol{s}| := \begin{cases} 0 & \text{if } \boldsymbol{s} = \epsilon \\ 1 + |\boldsymbol{t}| & \text{if } \boldsymbol{s} = \alpha \mathbin{.} \boldsymbol{t}.^{12} \end{cases}$$

Remark 1.65 (Languages as posets). Notice that we can define the following partial order on a language L over Σ:

- $\epsilon \preccurlyeq \boldsymbol{s}$ for all $\boldsymbol{s} \in L$, and
- $\boldsymbol{s} \prec \boldsymbol{t}$ if $\boldsymbol{t} = \alpha \mathbin{.} \boldsymbol{s}$ for some $\alpha \in \Sigma$.

One can verify that this is a ranked and well-founded partial order. Thus a version of theorem 1.52 adapted for $(L, \preccurlyeq)$ says we can define functions $f\colon L \to X$ for any set X by

$$f(\boldsymbol{s}) = \begin{cases} x & \text{if } \boldsymbol{s} = \epsilon \\ g(\boldsymbol{s}, f(\boldsymbol{t})) & \text{if } \boldsymbol{s} = \alpha \mathbin{.} \boldsymbol{t} \end{cases}$$

where $x \in X$ and $g\colon L \times X \to X$. This justifies definition 1.64(iv). (Notice that definition 1.64(iii) on the other hand, requires only theorem 1.41.)

Example 1.66. One often has to define functions on languages recursively, just as we did in definition 1.64. Let us give some more examples. Consider the alphabet $\Sigma = \{\mathtt{a}, \mathtt{b}\}$. We define the following functions for languages defined over this alphabet.

(i) has_a($\boldsymbol{s}$) should evaluate to **true** if $\boldsymbol{s}$ contains the symbol a, and false otherwise.

(ii) filt_a($\boldsymbol{s}$) returns the string $\boldsymbol{s}$ without any a's (if it had any).

The key here is to realise that every string drawn from Σ must be in one of the following forms: either the empty string ϵ, $\mathtt{a} \mathbin{.} \boldsymbol{t}$ for some string $\boldsymbol{t} \in \Sigma^*$, or $\mathtt{b} \mathbin{.} \boldsymbol{t}$ for some string $\boldsymbol{t} \in \Sigma^*$. We therefore give the following definitions.

[12] This time the formal analogue is a lot simpler, if $\boldsymbol{s}\colon k \to \Sigma$, then $|\boldsymbol{s}|$ is just k.

For has_a, clearly the empty string has no a. If a string is of the form a . $\boldsymbol{t}$, then it clearly does have an a. Otherwise, it is of the form b . $\boldsymbol{t}$, and has an a only if $\boldsymbol{t}$ has an a.

$$\text{has\_a}(\boldsymbol{s}) \coloneqq \begin{cases} \textbf{false} & \text{if } \boldsymbol{s} = \epsilon \\ \textbf{true} & \text{if } \boldsymbol{s} = \mathtt{a} \mathbin{.} \boldsymbol{t} \\ \text{has\_a}(\boldsymbol{t}) & \text{if } \boldsymbol{s} = \mathtt{b} \mathbin{.} \boldsymbol{t}. \end{cases}$$

Of course, this definition is justified by remark 1.65.

Now for filt_a, the empty string has no a so there is nothing to remove. Strings of the form a . $\boldsymbol{t}$ have the a removed, then we need to remove any a's which appear in $\boldsymbol{t}$. Finally, strings of the form b . $\boldsymbol{t}$ retain the b and have any a's from $\boldsymbol{t}$ removed.

$$\text{filt\_a}(\boldsymbol{s}) \coloneqq \begin{cases} \epsilon & \text{if } \boldsymbol{s} = \epsilon \\ \text{filt\_a}(\boldsymbol{t}) & \text{if } \boldsymbol{s} = \mathtt{a} \mathbin{.} \boldsymbol{t} \\ \mathtt{b} \mathbin{.} \text{filt\_a}(\boldsymbol{t}) & \text{if } \boldsymbol{s} = \mathtt{b} \mathbin{.} \boldsymbol{t}. \end{cases}$$

Continuing with the reasoning of remark 1.65, we can also obtain an induction theorem for languages. Indeed, applying theorem 1.56 to $L(\Sigma)$, we get the following.

Theorem 1.67 (Induction on languages)**.** *Let L be a language over Σ, and let $\Phi(\boldsymbol{s})$ be a statement about $\boldsymbol{s} \in L$. If*

(i) $\Phi(\epsilon)$ is true, and

(ii) for all $\boldsymbol{s} \in L$, if $\Phi(\boldsymbol{s})$ is true, then $\Phi(\alpha \mathbin{.} \boldsymbol{s})$ is true for each $\alpha \in \Sigma$,

then $\Phi(\boldsymbol{s})$ is true for all $\boldsymbol{s} \in L$.

When we apply theorem 1.56, we usually say we are performing induction on the "structure" of $a \in A$. In this case, when applying theorem 1.67, we say that we are performing induction on the structure of $\boldsymbol{s} \in L$.

Example 1.68. Let $\Sigma = \{\mathtt{a}, \mathtt{b}\}$, let $L \subseteq \Sigma^*$, and let has_a and filt_a be defined as in example 1.66. Intuitively, if we remove all the a's from a given string $\boldsymbol{s} \in L$, then it won't have any a's, that is,

$$\text{has\_a}(\text{filt\_a}(\boldsymbol{s})) = \textbf{false}$$

for any $\boldsymbol{s} \in L$. Let us prove this by induction on the structure of $\boldsymbol{s}$.

For the base case with $\boldsymbol{s} = \epsilon$, we have has_a(filt_a($\epsilon$)) = has_a($\epsilon$) = **false** by their definitions. Now for induction, suppose the result holds for some $\boldsymbol{t} \in \Sigma^*$, that is, assume has_a(filt_a($\boldsymbol{t}$)) = **false**. Then we have two cases to consider.

Case 1. $\boldsymbol{s} = \texttt{a} \,.\, \boldsymbol{t}$.

In this case

$$\begin{aligned}\text{has\_a}(\text{filt\_a}(\boldsymbol{s})) &= \text{has\_a}(\text{filt\_a}(\texttt{a} \,.\, \boldsymbol{t})) \\ &= \text{has\_a}(\text{filt\_a}(\boldsymbol{t})) && \text{(definition of filt\_a)} \\ &= \textbf{false}, && \text{(by the hypothesis)}\end{aligned}$$

as required.

Case 2. $\boldsymbol{s} = \texttt{b} \,.\, \boldsymbol{t}$.

In this case

$$\begin{aligned}\text{has\_a}(\text{filt\_a}(\boldsymbol{s})) &= \text{has\_a}(\text{filt\_a}(\texttt{b} \,.\, \boldsymbol{t})) \\ &= \text{has\_a}(\texttt{b} \,.\, \text{filt\_a}(\boldsymbol{t})) && \text{(definition of filt\_a)} \\ &= \text{has\_a}(\text{filt\_a}(\boldsymbol{t})) && \text{(definition of has\_a)} \\ &= \textbf{false}, && \text{(by the hypothesis)}\end{aligned}$$

as required. □

II. DETERMINISTIC FINITE-STATE AUTOMATA

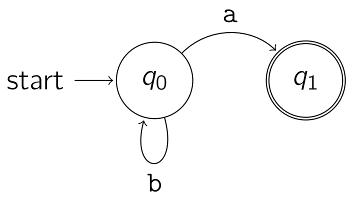
Here we introduce our first hypothetical "machine" for generating languages, called a deterministic finite-state automaton (DFSA). The idea behind a DFSA is that it is an object with "states" connected by arrows which we can travel along, and generate strings as we do so. For instance,

Definition 2.1 (Deterministic Finite State Automaton)**.** A *deterministic finite-state automaton* (DFSA) is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where Q is a finite set of states, Σ is a finite alphabet, $\delta : Q \times \Sigma \rightharpoonup Q$ is a (partial) function known as the *transition* function, $q_0 \in Q$ is the start state, and $F \subseteq Q$ is a finite set (not necessarily non-empty) of final/accept states.

Define $\delta_* : Q \times \Sigma^* \rightharpoonup Q$ to be a variant of the transition function δ which may be applied to strings. For all $q \in Q$, $\alpha \in \Sigma$ and $\boldsymbol{s} \in \Sigma^*$,

$$\delta_*(q, \boldsymbol{s}) := \begin{cases} q & \text{if } \boldsymbol{s} = \epsilon \\ \delta_*(\delta(q, \alpha), \boldsymbol{t}) & \text{if } \boldsymbol{s} = \alpha \,.\, \boldsymbol{t}. \end{cases}$$

Example 2.2. Consider the following DFSA.



As an example, we determine $\delta_*(q_0, \texttt{bba})$.

$$\begin{aligned} \delta_*(q_0, \texttt{bba}) &= \delta_*(\delta(q_0, \texttt{b}), \texttt{ba}) \\ &= \delta_*(q_0, \texttt{ba}) \\ &= \delta_*(\delta(q_0, \texttt{b}), \texttt{a}) \\ &= \delta_*(q_0, \texttt{a}) \\ &= \delta_*(\delta(q_0, \texttt{a}), \epsilon) \\ &= \delta_*(q_1, \epsilon) \\ &= q_1. \end{aligned}$$

Observe that similarly to δ, our function δ_* is a partial function (i.e. $\delta_*(q, \boldsymbol{s})$ does not necessarily exist for all $(q, \boldsymbol{s}) \in Q \times \Sigma^*$. For example, in the above automaton, $\delta_*(q_0, \texttt{aa})$ is undefined. Let us try and evaluate it:

$$\begin{aligned}\delta_*(q_0, \texttt{aa}) &= \delta_*(\delta(q_0, \texttt{a}), \texttt{a}) \\ &= \delta_*(q_1, \texttt{a}) \\ &= \delta_*(\delta(q_1, \texttt{a}), \epsilon),\end{aligned}$$

at which point we get stuck, since $\delta(q_1, \texttt{a})$ does not exist.

Lemma 2.3. *Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFSA, let $\boldsymbol{s}, \boldsymbol{t} \in \Sigma^*$, and let $q \in Q$. Then*

$$\delta_*(q, \boldsymbol{s} +\!\!+ \boldsymbol{t}) = \delta_*(\delta_*(q, \boldsymbol{s}), \boldsymbol{t}).$$

Proof. By induction on the structure of $\boldsymbol{s}$. For the base case with $s = \epsilon$, we have

$$\begin{aligned}\delta_*(q, \boldsymbol{s} +\!\!+ \boldsymbol{t}) &= \delta_*(q, \epsilon +\!\!+ \boldsymbol{t}) \\ &= \delta_*(q, \boldsymbol{t}) && (\text{since } \epsilon +\!\!+ \boldsymbol{\sigma} = \boldsymbol{\sigma} \text{ for any } \boldsymbol{\sigma} \in \Sigma^*) \\ &= \delta_*(\delta_*(q, \epsilon), \boldsymbol{t}), && (\text{by definition of } \delta_*)\end{aligned}$$

as required. Now we proceed with the inductive step. Suppose the result holds for $\boldsymbol{s}' \in \Sigma^*$. We show it holds for $\boldsymbol{s} = \alpha \mathbin{.} \boldsymbol{s}'$, where $\alpha \in \Sigma$.

We consider two different cases. First, suppose $\delta(q, \alpha)$ is defined. Then

$$\begin{aligned}\delta_*(q, \boldsymbol{s} +\!\!+ \boldsymbol{t}) &= \delta_*(q, \alpha \mathbin{.} \boldsymbol{s}' +\!\!+ \boldsymbol{t}) \\ &= \delta_*(\delta(q, \alpha), \boldsymbol{s}' +\!\!+ \boldsymbol{t}) && (\text{by definition of } \delta_*) \\ &= \delta_*(\delta_*(\delta(q, \alpha), \boldsymbol{s}'), \boldsymbol{t}) && (\text{by the hypothesis}) \\ &= \delta_*(\delta_*(q, \alpha \mathbin{.} \boldsymbol{s}'), \boldsymbol{t}) && (\text{by definition of } \delta_*) \\ &= \delta_*(\delta_*(q, \boldsymbol{s}), \boldsymbol{t}),\end{aligned}$$

as required. If, on the other hand, $\delta(q, \alpha)$ is undefined, then the left-hand side is

$$\begin{aligned}\delta_*(q, \boldsymbol{s} +\!\!+ \boldsymbol{t}) &= \delta_*(q, \alpha \mathbin{.} \boldsymbol{s}' +\!\!+ \boldsymbol{t}) \\ &= \delta_*(\delta(q, \alpha), \boldsymbol{s}' +\!\!+ \boldsymbol{t}), && (\text{by definition of } \delta_*)\end{aligned}$$

which is undefined, and the right-hand side is

$$\begin{aligned}\delta_*(\delta_*(q, \boldsymbol{s}), \boldsymbol{t}) &= \delta_*(\delta_*(q, \alpha \mathbin{.} \boldsymbol{s}), \boldsymbol{t}) \\ &= \delta_*(\delta_*(\delta(q, \alpha), \boldsymbol{s}), \boldsymbol{t}), && (\text{by definition of } \delta_*)\end{aligned}$$

which is also undefined. Thus if $\delta(q, \alpha)$ is undefined, then both sides of the result are undefined.

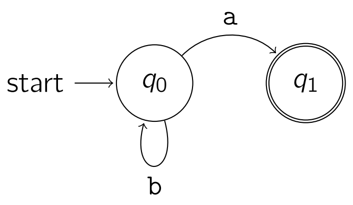
Therefore in both cases, the result holds. $\square$

Note that it is important to consider the case where $\delta(q, \alpha)$ is undefined separately. Otherwise we cannot sensibly tackle the case where it is defined.

Definition 2.4 (Accepted Language)**.** Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFSA. Then $\mathcal{L}(M)$ denotes the *language accepted by* the automaton M, given by

$$\mathcal{L}(M) = \{\boldsymbol{s} \in \Sigma^* : \delta_*(q_0, \boldsymbol{s}) \in F\}.$$

Example 2.5. Consider the DFSA from earlier, which we shall call M.



We claim $\mathcal{L}(M) = \{\mathtt{b}^n \mathbin{+\!\!+} \mathtt{a} : n \geqslant 0\}$.

Proof. As usual for sets, we prove that each is a subset of the other. First we show that $\{\mathtt{b}^n \mathbin{+\!\!+} \mathtt{a} : n \geqslant 0\} \subseteq \mathcal{L}(M)$. In other words, we need that $\delta_*(q_0, \mathtt{b}^n \mathbin{+\!\!+} \mathtt{a}) \in F$ for all $n \geqslant 0$. This easily follows by induction on n. For the base case, $n = 0$ gives $\delta(q_0, \mathtt{b}^0 \mathbin{+\!\!+} \mathtt{a}) = \delta(q_0, \epsilon \mathbin{+\!\!+} \mathtt{a}) = \delta(q_0, \mathtt{a}) = q_1 \in F$. For the step, suppose the result holds for $n = k \in \mathbb{N}$. We show that it holds for $n = k + 1$:

$$\begin{aligned}
\delta_*(q_0, \mathtt{b}^{k+1} \mathbin{+\!\!+} \mathtt{a}) &= \delta_*(q_0, \mathtt{b} \,.\, \mathtt{b}^k \mathbin{+\!\!+} \mathtt{a}) && \text{(by definition of string exp)}\\
&= \delta_*(\delta(q_0, \mathtt{b}), \mathtt{b}^k \mathbin{+\!\!+} \mathtt{a}) && \text{(by definition of } \delta_*\text{)}\\
&= \delta_*(q_0, \mathtt{b}^k \mathbin{+\!\!+} \mathtt{a}) && \text{(by evaluating } \delta(q_0, \mathtt{b})\text{)}\\
&\in F, && \text{(by the hypothesis)}
\end{aligned}$$

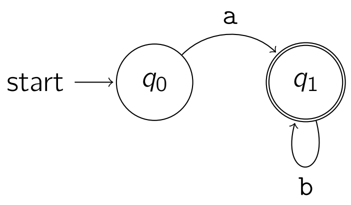
as required. Now we show that $\mathcal{L}(M) \subseteq \{\mathtt{b}^n \mathbin{+\!\!+} \mathtt{a} : n \geqslant 0\}$. First of all, notice that the only possible final state is q_1. So it must be the case that if $\delta_*(q_0, \boldsymbol{s})$ is defined (i.e., we have $\delta_*(q_0, \boldsymbol{s}) \in F$), then $\delta_*(q_0, \boldsymbol{s}) = q_1$. So we are claiming that whenever this is the case, $\boldsymbol{s} = \mathtt{b}^n \mathbin{+\!\!+} \mathtt{a}$ for some $n \geqslant 0$.

We prove this by induction on the structure of $\boldsymbol{s}$. With $\boldsymbol{s} = \epsilon$, we have $\delta_*(q_0, \boldsymbol{s}) = q_0$, so $\epsilon \notin F$, and hence the implication is vacuously true. Now suppose the result holds for some $\boldsymbol{s}' \in \Sigma^*$ such that $\delta_*(q_0, \boldsymbol{s}') = q_1$. We show it holds for $s = \alpha \,.\, \boldsymbol{s}'$. If $\alpha = \mathtt{a}$, then we have $\delta_*(q_0, \alpha \,.\, \boldsymbol{s}') = \delta_*(\delta(q_0, \alpha), \boldsymbol{s}') = \delta_x(q_1, \boldsymbol{s}')$.

(strengthen the IH) □

Remark 2.6. Observe that in example 2.5, it was necessary to strengthen the inductive hypothesis to prove a weaker form of the statement.

Example 2.7. Let M be the automata depicted below.



We show that $\mathcal{L}(M) = \{\mathtt{a} +\!\!+ \mathtt{b}^n : n \geqslant 0\}$.

Proof. First we show that □