

CM10196 Topic 4: Functions and Relations

Guy McCusker

1W2.1

Functions and relations

Perhaps the most widely used notion in all of mathematics is that of *function*.

Informally, a function is an “operation” which takes an input value and gives an output value. We can (and will) be more precise about this.

From the computer science perspective, functions play a vital role: much of what we do when programming is writing code to compute a particular function.

Relations

A more general notion than function is that of *relation*.

Roughly speaking, a relation is something which provides an association between elements of sets.

For instance, on the set of all people there is the relation “is in the same family as”. This relation *relates* two people (two elements of the set) exactly when those people are “related” in the usual informal sense.

Cartesian product of sets

To make a precise definition of a relation, and later of a function, we will make use of the notion of *Cartesian product of sets*.

The Cartesian product is used to talk about *pairs* of things, one from one set and one from another.

Definition

The *Cartesian product* of sets A and B , written $A \times B$, is defined as the set of pairs

$$\{(x, y) \mid x \in A \wedge y \in B\}.$$

The notation (x, y) means a pair of elements: we say that $(x, y) = (z, w)$ if and only if $x = z$ and $y = w$.

Cartesian product

Example

- 1 If $A = \{a, b, c\}$ and $B = \{0, 1\}$ then $A \times B$ is the set

$$\{(a, 0), (a, 1), (b, 0), (b, 1), (c, 0), (c, 1)\}.$$

- 2 If $A = \{x \mid x \text{ is a real number, } 0 \leq x \leq 1\}$ and $B = \{x \mid x \text{ is a real number, } 1 \leq x \leq 2\}$ then $A \times B$ can be thought of as a rectangle in the plane whose corners (vertices) have coordinates $(0, 1)$, $(0, 2)$, $(1, 1)$ and $(1, 2)$.
- 3 If A is the set of students at Bath and B is the set of candidate numbers, then $A \times B$ is the set of *all possible* student-candidate number pairs. The *actual* mapping from students to their real candidate numbers is a subset of this.

The idea of a mapping being a subset of the Cartesian product is what will underlie our formal definitions of relation and function.

Definition alert!

You should by now be jumping up and down and panicking about the possibility that the "notation" (a, b) for pairs of things is not well-defined.

Well, perhaps not, but it is a question worth asking. What do we mean by such a pair?

We certainly do *not* mean a two-element set: the pairs $(0, 1)$ and $(1, 0)$ are supposed to be different, while the sets $\{0, 1\}$ and $\{1, 0\}$ are the same.

Remember that for sets, the vital ingredient was that sets are equal when they contain the same elements

For pairs, the key is that pairs are equal when they contain the same elements, listed in the same order. That is,

$$(a, b) = (c, d) \equiv a = c \wedge b = d.$$

Formal definition of pairs

It is probably reasonable simply to accept that such pairs exist.

However, if we were doing formal foundations of mathematics, we'd have to make some kind of precise definition like:

Definition

given elements a and b , the pair (a, b) is defined to be the set $\{\{a\}, \{a, b\}\}$.

If you are so inclined, you can check that this definition satisfies the requirement we laid out for pairs.

Relations

Given two sets A and B , a *relation between A and B* is a subset of $A \times B$.

Example

- 1 If P is the set of all people, and D is the set of days of the year, then the set of pairs of the form (person, person's birthday) is a relation between P and D .
- 2 The set of pairs of the form (person, child of person) is a relation between P and itself. This sort of thing is called a *binary relation on P* .
- 3 The relation $<$ is a binary relation on \mathbb{Z} : it is defined by the set of pairs

$$\{(x, y) \mid x \in \mathbb{Z}, y \in \mathbb{Z}, x < y\}.$$

We usually write $x < y$ instead of $(x, y) \in <$, of course.

- 4 The relation of equality, $=$, is a binary relation on any set. On the integers, for example, it is the set of pairs $\{(x, x) \mid x \in \mathbb{Z}\}$.

Notation

The last two examples introduce a useful piece of notation: given a relation R between A and B , we often write $a R b$ to mean that $(a, b) \in R$.

Read $a R b$ as “ a is related to b by R ” or something similar.

For the case of relations like $<$, \leq , $=$ and so on, this notation is exactly what we are used to.

The relation of equality on any set A is called the *identity relation* because it relates identical things and nothing else.

Directed graphs

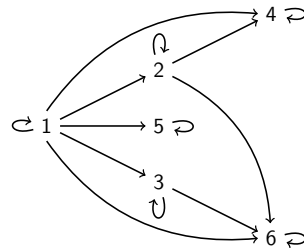
A *directed graph* is a structure containing:

- ▶ a set V of *vertices* (also called *nodes*)
- ▶ a set A of *arcs* (also called *edges*) where each arc connects one vertex to another, and the direction of the arc is important. The vertex where an arc starts is its *source*; the vertex where it ends is its *target*.

Small graphs can readily be drawn as collections of *dots* (for the vertices) and *arrows* between the dots (for the arcs).

Directed graphs

Here's a picture of a directed graph. In this case, the nodes are the numbers from 1 to 6, and there is an arc from one vertex to another if and only if the first number divides the second.



Directed graphs are relations

This kind of directed graph is really the same as a binary relation on a set:

- ▶ the set of vertices is the set on which the relation works
- ▶ the set of arcs defines a set of pairs (a, b) where a is the source vertex of an arc and b is the target vertex.

The picture we drew on the previous slide describes the relation

$$\left\{ \begin{array}{l} (1, 1), (2, 2), (3, 3), (4, 4), (5, 5), (6, 6), (1, 2), \\ (1, 3), (1, 4), (1, 5), (1, 6), (2, 4), (2, 6), (3, 6) \end{array} \right\}$$

which we could alternatively define as

$$\{(a, b) \mid a, b \in \mathbb{Z}, 1 \leq a \leq 6, 1 \leq b \leq 6, a \text{ divides } b\}.$$

Other kinds of graphs

While we're talking about graphs, we should mention that there are other kinds of graphs:

- ▶ undirected graphs, where the direction of an arc doesn't matter
- ▶ labelled graphs, where arcs are labelled with some information
- ▶ graphs in which there can be more than one arc from one vertex to another
- ▶ ...

Graphs in computing

Graphs are an important concept in computer science.

They're useful for modelling a great many things, for example road or rail networks: a vertex is a place, and an arc is a route from one place to another.

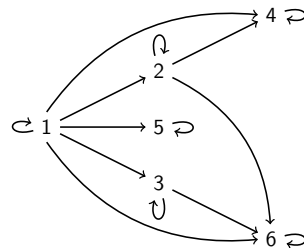
If we label the arcs with information about the time it takes to travel along an arc, then we can ask questions like

- ▶ "how long does it take to get from Baker Street to Mornington Crescent?"
- ▶ "what's the best way to Turnham Green?"

There are powerful algorithms for discovering this kind of information from such graphs. If you're interested, look up Dijkstra's Algorithm.

Classifying relations

Looking at the picture of the divisibility relation we drew above:



we notice that every node has a "loop" attached to it: that is, every element of the set is related to itself.

This is a special property of certain binary relations, called *reflexivity*.

Classifying relations

Definition

A binary relation R on a set A is *reflexive* if $a R a$ for every $a \in A$.

That is to say, "every node has a loop attached." Examples of reflexive relations include

- ▶ " a divides b ", a relation on the integers
- ▶ " $a \leq b$ ", also on the integers
- ▶ " a lives at the same address as b ", on the set of people
- ▶ the identity relation on any set.

Other relations, like $<$, never relate an element to itself:

Definition

A binary relation R is called *irreflexive* if we do not have $a R a$ for any $a \in A$.

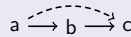
Classifying relations

Another property which some relations possess is *transitivity*.

On graphs, this is the property that if you can get from a to b (i.e. there's an arc connected a to b) and from b to c , then there's an arc from a to c .

Definition

A binary relation R on a set A is called *transitive* if whenever $a R b$ and $b R c$, we also have $a R c$. In a picture:



Whenever the solid arcs exist, the dashed arc must also exist.

Examples

The following relations are transitive:

- ▶ " a divides b "
- ▶ $<$, $=$ and \leq
- ▶ " a is younger than b " on the set of people
- ▶ " a is an ancestor of b " on the set of people

The following are not transitive:

- ▶ " a is within 5 minutes' drive of b " on the set of bus stops on the Number 18 route.
- ▶ " a beat b at home in the 2006-7 football season", on the set of premiership football teams.
- ▶ " $a^2 = b$ " on the set of integers.

Classifying relations

Another key property is *symmetry*: on graphs, this is the property that if you can get from a to b , you can also get back from b to a .

(Think of the difference between one-way and two-way streets.)

Definition

A binary relation R on a set A is *symmetric* if whenever $a R b$ we also have $b R a$.



Examples

The following relations are symmetric:

- ▶ $=$, the identity relation on any set
- ▶ "a is married to b" on the set of people

These ones are not:

- ▶ $<$ and \leq
- ▶ "a is younger than b" on the set of people
- ▶ "a is an ancestor of b" on the set of people

Question: do you expect that the relation "a is within five minutes' drive of b" on number 18 bus stops is symmetric?

Classifying relations

Another property, a bit like "never having symmetry", is *antisymmetry*.

Definition

A relation R on a set A is *antisymmetric* if, whenever $a R b$ and $b R a$, it is the case that $a = b$.

This says that the only entries in the relation which can be "flipped around" to yield a new entry in the relation are those of the form (a, a) .

Examples:

- ▶ \leq is antisymmetric: if $a \leq b$ and $b \leq a$ then $a = b$.
- ▶ $<$ is also antisymmetric: if $a < b$ then we never have $b < a$ so there's nothing to check.

Equivalence relations

We've seen that equality is a relation (on any set).

We can generalize this to capture those relations which behave like equality. They are called *equivalence relations*.

Definition

A binary relation R on a set A is an equivalence relation if it is

- ▶ reflexive
- ▶ transitive, and
- ▶ symmetric.

This says that a “notion of equivalence” should satisfy

- ▶ every element is equivalent to itself
- ▶ if a is equivalent to b then b is equivalent to a
- ▶ if a is equivalent to b and b is equivalent to c then a is equivalent to c .

Example

A key example of equivalence relation is “equality modulo p ” for some number p .

Given integers m , n and p , we define $m = n \pmod{p}$ if there are integers a , b and c , with $0 \leq c < p$ such that

$$m = a \times p + c \quad n = b \times p + c.$$

Exercise

Prove that this is indeed an equivalence relation.

Equivalence relations in computing

Equivalence relations play an important role in computing, in many ways. One way is this.

When programming, we often set up data structures to represent information that we're recording. For instance, we might use an array to collect together the names of students on a course.

What we really care about here is the *set* of names recorded in the array: the order does not matter.

So multiple different arrays can represent the same information. We can capture this notion of “representing the same information” with an equivalence relation.

Equivalence classes

We often use symbols like \approx for equivalence relations.

Given a set A with an equivalence relation \approx , for every element $a \in A$ we define the *equivalence class of a* , written $[a]$, as the set

$$\{b \in A \mid b \approx a\}.$$

Notice that we always have $a \in [a]$ and if $a \approx b$ then $[a] = [b]$.

(Check that these facts really are facts.)

Equivalence classes: example

Let S be the set of students on this course.

Define an equivalence relation \approx such that $a \approx b$ if and only if a and b are in the same tutor group.

Then $[a]$ is the set of students in the same tutor group as a .

Other examples:

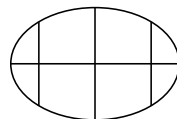
- ▶ for the identity relation, the equivalence class of an element a is the singleton set $\{a\}$.
- ▶ for the relation “equality modulo 2” on the integers, the equivalence class $[0]$ is all even numbers, and the equivalence class $[1]$ is all odd numbers.

Equivalence classes are partitions

Given a set A , a set S of subsets of A is called a *partition* of A if:

- ▶ every pair of sets in S is disjoint: if $P_1 \in S$ and $P_2 \in S$ with $P_1 \neq P_2$ then $P_1 \cap P_2 = \emptyset$
- ▶ every element of A is contained in some $P \in S$.

That is to say, the sets in S split the whole of A up into chunks.



Equivalence classes and partitions

The set of equivalence classes of an equivalence relation gives us a partition on the set; and any partition generates an equivalence relation.

Theorem

If \approx is an equivalence relation on A then the set of equivalence classes

$$\{[a] \mid a \in A\}$$

forms a partition of A .

Conversely, if S is a partition of A , then the relation \approx defined by

$$\{(a, b) \mid \exists P \in S. a \in P \wedge b \in P\}$$

is an equivalence relation, whose equivalence classes are exactly the members of S .

Proof of the theorem

We'll prove the first part of the theorem.

Given an equivalence relation \approx on A , we have to show that the set of equivalence classes of \approx partitions A .

Suppose $[a] \cap [b]$ is non-empty. Then there is some $c \in [a] \cap [b]$, so that $c \approx a$ and $c \approx b$. By symmetry and transitivity, $a \approx b$, so $[a] = [b]$.

For any $a \in A$, $a \in [a]$.

That's all we needed to show!

Quotient of a set

Given a set A and equivalence relation \approx on A , the set of equivalence classes is called the *quotient of A by \approx* and is written A/\approx .

Its importance for computing is, among other things, the notion of data representation we mentioned before.

If A is our set of representations, and \approx is the correct notion of equivalence, then A/\approx is (a good representation of) the set of things we're encoding.

Example: representing integers using naturals

In previous lectures we discussed how to set up the theory of the natural numbers using Peano arithmetic.

In this development, the number 3 is represented as $S(S(S(0)))$, for example.

Once we've done this, and defined operations like $+$ and so on, we can go on to represent the integers.

One way to represent an integer is as a pair (m, n) of natural numbers; we think of this as representing the integer $m - n$.

This of course means that a given integer has infinitely many representations: -3 is represented by any of

$$(0, 3) \quad (1, 4) \quad (2, 5) \quad (3, 6) \quad \dots$$

Why use this representation?

This representation has its benefits, though.

For example, it is easy to define addition on integers using natural number addition:

$$(m_1, n_1) + (m_2, n_2) = (m_1 + m_2, n_1 + n_2).$$

It is also easy to define multiplication by -1 :

$$(m, n) \times -1 = (n, m)$$

and hence subtraction

$$(m_1, n_1) - (m_2, n_2) = (m_1 + n_2, n_1 + m_2).$$

Exercise

Define multiplication of integers in this representation.

An equivalence relation

Since our representation gives infinitely many ways of encoding an integer, we need a way of telling when two such encodings are the same integer.

We can define an equivalence relation

$$(m_1, n_1) \approx (m_2, n_2) \equiv m_1 + n_2 = m_2 + n_1.$$

Now two encodings of the same integer are related by \approx , so there is a close correspondence between integers and *equivalence classes* of representations. That is, we can think of $\mathbb{N} \times \mathbb{N} / \approx$ as being the set of integers.

We'll say a little more about this later.

Partial orders

Another common form of relation is the *ordering relation*, like \leq .

Definition

A binary relation R on a set A is called a *partial order* if it is

- ▶ reflexive
- ▶ transitive, and
- ▶ antisymmetric.

This says that a “notion of ordering” should satisfy

- ▶ every element is less than or equal to itself
- ▶ if a is less than or equal to b and b is less than or equal to c then a is less than or equal to c
- ▶ if a is less than or equal to b and b is less than or equal to a then $a = b$.

Examples

The following relations are partial orders:

- ▶ \leq on the integers
- ▶ $=$ on any set
- ▶ \subseteq on the set of subsets of some set A (that is, the powerset $\mathcal{P}(A)$).

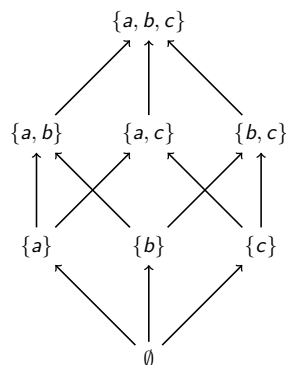
Note that $<$ is not a partial order: it is not reflexive. Our notion of partial order is “non-strict”, that is, it generalizes “less than or equal”-style orderings.

Exercise

Define a notion of “strict partial order”, which generalizes $<$ -style orderings.

The subset relation as a graph

Omitting the reflexive and transitive arcs, the subset relation on $\mathcal{P}\{a, b, c\}$ looks like this:



Composition of relations

Consider the relations “is a parent of” and “is a grandparent of”. It should be clear that we can define the grandparent relation from the parent relation.

A person a is a grandparent of person c if

- ▶ a is a parent of somebody, b , and
- ▶ b is a parent of c .

More generally, if we have a relation R between A and B , and a relation S between B and C , we can define a relation $S \circ R$ by

$$\{(a, c) \mid \exists b \in B. a R b \wedge b S c\}.$$

The “grandparent” relation is

$$\text{“parent”} \circ \text{“parent”}.$$

Inverse

Given a relation R between A and B , we can define a relation R^{-1} between B and A by

$$\{(b, a) \mid a R b\}.$$

That is, we just flip around the relation R to get R^{-1} .

Symmetry and transitivity again

Theorem

A binary relation R on a set A is symmetric if and only if $R = R^{-1}$.

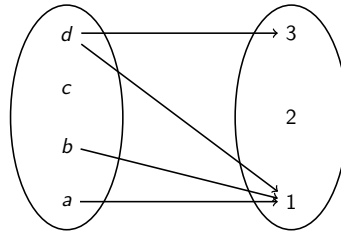
R is transitive if and only if $R \circ R \subseteq R$.

Exercise

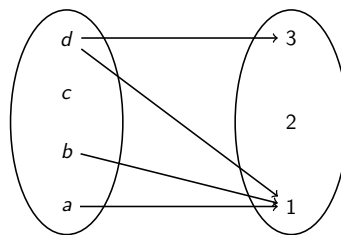
Prove this theorem!

Picturing relations between sets

We might draw a relation from one set to another like this:



Observations

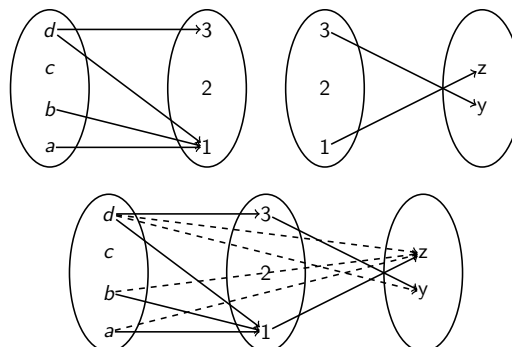


Notice that:

- ▶ not every element of the left hand set is related to something on the right (i.e. is the source of an arc)
- ▶ some elements of the left hand set are related to more than one thing on the right (i.e. they are the source of more than one arc)
- ▶ not every element of the right hand set is the target of an arc
- ▶ some elements are the target of more than one arc.

Picturing composition

Let's see what the composition of two relations looks like.



Functions

Definition

A function f from a set A to a set B is a relation such that every $a \in A$ is related to exactly one $b \in B$. That is:

- ▶ $\forall a \in A. \exists b \in B. (a, b) \in f$, and
- ▶ $\forall a \in A. \forall b_1, b_2 \in B. (a, b_1) \in f \wedge (a, b_2) \in f \rightarrow b_1 = b_2$.

When f is a function from A to B we write $f : A \rightarrow B$ and use the notation $f(a) = b$ to mean that $(a, b) \in f$.

Functions

Thus a function is a relation such that every element of the left-hand set A is the source of exactly one arc to an element of B .

A function is said to *map* the elements of A into B , and so functions are also called *maps* or *mappings*.

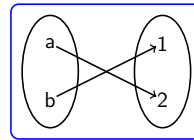
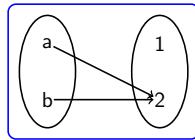
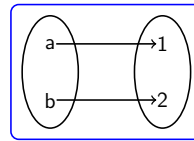
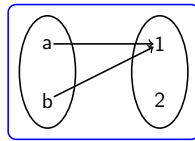
It is still possible that an element of B may be the target of no arcs, or more than one.

Examples

- ▶ Most of the usual operations on numbers and so on are functions: addition is a function from $\mathbb{Z} \times \mathbb{Z}$ to \mathbb{Z} for example.
- ▶ The identity relation on any set is a function from that set to itself. We call it the *identity function*.
- ▶ The relation which connects students to their candidate numbers is a function from the set of all students to the set of all candidate numbers.
- ▶ The relation connecting students taking CM10196 to their marks on problem sheet 4 is a function from the set of students to the set $\{0, 1, 2, \dots, 10\}$.
- ▶ Many computer programs compute a function from the set of possible inputs to the set of possible outputs. Not all programs do, though. Why not?

Examples

There are four possible functions from the set $\{a, b\}$ to the set $\{0, 1\}$.



Cardinality, product and function space

For a finite set A , define $|A|$, the *cardinality* of A , to be the number of elements in the set A .

Theorem

Given sets A and B , $|A \times B|$ is $|A| \times |B|$.

The set of all functions from A to B has cardinality $|B|^{|A|}$

Partly motivated by this, the set of functions from A to B is sometimes written as B^A .

Composition of functions

Since functions are relations, we can compose them just like relations.

Given $f : A \rightarrow B$ and $g : B \rightarrow C$, the composite $g \circ f : A \rightarrow C$ is

$$\{(a, c) \mid \exists b \in B. f(a) = b \wedge g(b) = c\}.$$

That is, $(g \circ f)(a) = g(f(a))$.

Example

If f is the map taking x to $x + 1$ and g is the map taking x to x^2 (so that both f and g go from \mathbb{Z} to itself), then $g \circ f$ is the function taking x to $(x + 1)^2$.

Inverses

Since a function f is a special relation, its inverse, the relation f^{-1} , always exists.

Normally, though, when we say that a function has an inverse, we mean that the inverse is itself a function.

Under what circumstances does the inverse relation f^{-1} give us a function?

It is the relation

$$\{(b, a) \mid f(a) = b\}.$$

For this to be a function, we need two properties:

- ▶ every b must be sent somewhere, i.e. for every $b \in B$ there must exist some $a \in A$ such that $f(a) = b$.
- ▶ every b must be sent to exactly one place, i.e. if $f(a_1) = b$ and $f(a_2) = b$ then $a_1 = a_2$.

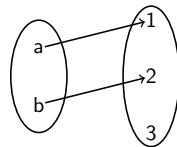
These two special properties of f are very important.

Surjective functions

Definition

A function $f : A \rightarrow B$ is called *surjective*, or “onto”, if for every $b \in B$ there is some $a \in A$ with $f(a) = b$.
Such a function is also called a *surjection*.

The condition says that the function “hits everything” in B . That is to say, we do not have this kind of situation:



Nothing “hits” 3 in this example.

Examples

- ▶ The function taking a number x to $x + 1$ is surjective as a function from \mathbb{Z} to \mathbb{Z} . It is not surjective as a function from \mathbb{N} to \mathbb{N} .
- ▶ The function taking x to x^3 is surjective on the real numbers: every real number has a cube root.
- ▶ For any route on the London Underground (i.e. a chosen line, and a chosen direction of travel) we can define a function taking a station to the next station on the route. These functions are not usually surjective (think about the first station on the route) except for the Circle Line, which goes round in a circle.
- ▶ On the set $\{0, 1, 2, 3, 4\}$, the function which takes x to $2 \times x \bmod 5$ is surjective.

Exercise

Check that the last claim is true!

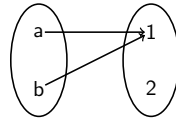
Injective functions

Definition

A function $f : A \rightarrow B$ is called *injective*, or 1-1, if whenever $f(a_1) = f(a_2)$, we have $a_1 = a_2$.

We also say that f is an *injection*.

The condition says that no two distinct elements of A are mapped to the same place, i.e. we don't have this sort of picture:



Examples of injections

The following functions are injective:

- ▶ the identity function on any set
- ▶ the function taking any integer x to $x + 1$
- ▶ the function taking any integer x to 2^x
- ▶ the function taking a natural number x to x^2 is injective;

Note that the function taking x to x^2 is *not* injective when seen as a function on integers, because (for example), $(-1)^2 = 1 = 1^2$.

Inverse again

Now we can state a theorem:

Theorem

Given a function $f : A \rightarrow B$, the relation f^{-1} is itself a function if and only if f is both injective and surjective.

Functions which are both injective and surjective are so special that they have a special name: they are called *bijective*.

Some examples:

- ▶ the function mapping x to $-x$ on the integers is bijective
- ▶ the function mapping x to x^3 on the real numbers is bijective
- ▶ the function mapping n to $2 \times n$ is a bijection from the integers to the even integers
- ▶ the function mapping n to $2 \times n \bmod 2$ is a bijection from $\{0, 1, 2, 3, 4\}$ to itself
- ▶ the identity map on any set is bijective.

Some theorems

Theorem

Let $f : A \rightarrow B$ and $g : B \rightarrow C$.

- ▶ if f and g are both injective, then $g \circ f$ is injective
- ▶ if f and g are both surjective, then $g \circ f$ is surjective
- ▶ if f and g are both bijective, then $g \circ f$ is bijective (this follows from the previous two facts)
- ▶ if f is bijective so is f^{-1}
- ▶ if $g \circ f$ is injective then f is injective
- ▶ if $g \circ f$ is surjective then g is surjective

Proof of one of these facts

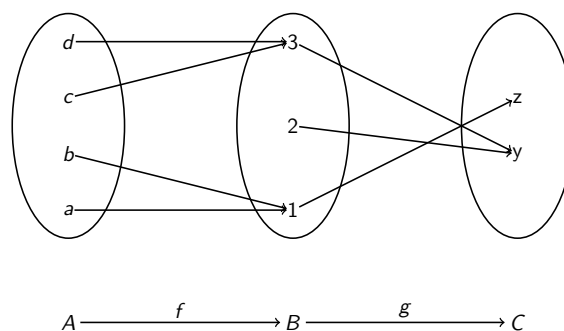
Let's prove the last of these: if $g \circ f$ is surjective then g is surjective.

Suppose that $g \circ f$ is surjective, and try to show that g is surjective, that is, that for any $c \in C$ we can find some $b \in B$ such that $g(b) = c$.

Since $g \circ f$ is surjective, there is some $a \in A$ such that $(g \circ f)(a) = c$, that is, $g(f(a)) = c$, so letting $b = f(a)$ establishes what we need.

Remark

If $g \circ f$ is surjective, it does not necessarily follow that f is surjective, as the picture below shows:



Functions and equivalence relations

Remember our representation of the integers using pairs of naturals: a pair (m, n) represents the integer $m - n$.

We defined an equivalence relation on pairs: $(m_1, n_1) \approx (m_2, n_2)$ if and only if

$$m_1 + n_2 = m_2 + n_1.$$

We then said two things:

- ▶ the equivalence classes behave like integers
- ▶ we can define functions on these pairs, like subtraction:

$$(m_1, n_1) - (m_2, n_2) = (m_1 + n_2, n_1 + m_2).$$

Do these things really make sense together?

Functions on equivalence classes

If we've got a set A with an equivalence relation \approx , and a function $f : A \rightarrow A$, we'd sometimes like to use f as a function on *equivalence classes*.

That is, we'd like to define a function g from A/\approx to itself by

$$g([a]) = [f(a)].$$

The question is, does this make sense?

Functions that respect equivalence

In order for this to define a function, it has to be the case that

$$a_1 \approx a_2 \rightarrow f(a_1) \approx f(a_2).$$

When $a_1 \approx a_2$ we have $[a_1] = [a_2]$, so we need $[f(a_1)] = [f(a_2)]$ or g will not be a function.

If f has this property we say that f respects \approx .

The vital fact about our operations on the "integers as pairs" representation is that they all respect the equivalence relation in this way.

Subtraction respects \approx

Let's verify that subtraction respects \approx .

Suppose $(m_1, n_1) \approx (m'_1, n'_1)$ and $(m_2, n_2) \approx (m'_2, n'_2)$. We shall show that

$$(m_1, n_1) - (m_2, n_2) \approx (m'_1, n'_1) - (m'_2, n'_2).$$

This is just a question of unpacking definitions. By definition

$$(m_1, n_1) - (m_2, n_2) = (m_1 + n_2, n_1 + m_2)$$

$$(m'_1, n'_1) - (m'_2, n'_2) = (m'_1 + n'_2, n'_1 + m'_2)$$

so we have to check that these things are equivalent, i.e. that

$$m_1 + n_2 + n'_1 + m'_2 = m'_1 + n'_2 + n_1 + m_2.$$

But we know that

$$m_1 + n'_1 = n_1 + m'_1 \quad \text{and} \quad m_2 + n'_2 = n_2 + m'_2$$

so the required fact follows.