Mathematical Preliminaries

CSCI 2210

last updated September 6, 2023

In order to give a formal account of computation we will need some mathematical tools. Most of these are standard from a discrete maths course. If they are new to you, don't panic and don't feel intimidated. Just try to absorb as much of the main ideas as you can now and then come back to them as they arise later in the course. After you've applied them a few times they will start to feel familiar and comfortable, and it will save us time and tedium in the future when we can all just mentally `import monoid` or something, rather than having to describe each instance we encounter on a case-by-case basis.

1 Sets

We begin with the concept of a set. The idea is that a *set* is a collection of things. What are the things, you may ask, and in what ways are we allowed to collect them? These turn out to be deep questions that we will side-step as much as possible for now. I am tempted to begin with the concept of *type* rather than *set*, but that would probably take us too far afield.

All we need to know about sets is that they are completely determined by their members: some things are in, some are out, and a set is expected to know which is which. We indicate that a thing x is an *element* (synonym: "member") of a set A by writing "$x \in \mathrm{A}$". Fun fact: the symbol "$\in$" is a stylized ε (Greek letter epsilon), which is mnemonic for the word "element".

We can specify a set by specifying its elements. It is traditional to use brace-notation to do this. Some sets are quite small and we can just list their elements: $\text{Beatles} = \{\text{John}, \text{Paul}, \text{George}, \text{Ringo}\}$. Some, like the *natural numbers*, are infinite, so we can't enumerate them and must specify their members by other means: $\mathbb{N} = \{0, 1, 2, 3, \cdots\}$. Some are even infinite-r, like the real numbers, and require some ingenuity in order to describe their members. We will deal mostly with sets that are either finite, or countably infinite like $\mathbb{N}$.

Because sets are characterized entirely by the element relationship, the order in which we specify their members does not matter, nor does any potential repetition: once you're in, you're in, and that's all there is to it. Two sets are *equal* just in case they have the same elements.

From the element relationship between things and sets we can define an important relationship between two sets, the subset relationship. The set B is a *subset* of the set A, written "$\mathrm{B} \subseteq \mathrm{A}$" if every element of B is also an element of A:

$$\mathrm{B} \subseteq \mathrm{A} \quad :=\quad \forall\, x \in \mathrm{B}\,.\, x \in \mathrm{A}.$$

Here the ":=" symbol can be read as "is defined as", or "means", the "$\forall$" symbol means "for all" or "for each", and the dot indicates that we are about to make a statement about the things in question. So the definition of B being a subset of A is that for each element x in B it is the case that x is an element of A. You may have seen this expressed in the form of a *Venn diagram* where the circle for B is contained within the circle for A.

The *cardinality* of a set is the number of elements that it contains. It is customary to indicate this using vertical bars: $|\text{Beatles}| = 4$. The smallest possible set has cardinality 0. It is called the *empty set* and contains no elements: $\emptyset := \{\,\}$. The next smallest cardinality for a set is 1. A set with just one element is called a *singleton*. There are as many singleton sets as there are things. If we want to refer to an arbitrary but fixed singleton set we will use $\mathbb{1} := \{\star\}$. An important two element set the *Booleans*, $\mathbb{2} := \{\text{true}, \text{false}\}$. Of course we could continue naming sets like this, but we won't bother. Note that while it is sound to postulate the

existence of a set that contains *nothing*, it turns out to be problematic to postulate the existence of a set that contains *everything*, which leads to Russell's paradox.

Every set has the empty set as a subset. We will now state and prove this formally. We do this in the form of a *theorem*, which is just an assertion that we back up with a proof that it is true.

Theorem 1
Every set has the empty set as a subset.

Proof. Let C be an arbitrary set. By the definition of subset, the empty set is a subset of C, i.e. $\emptyset \subseteq \mathrm{C}$, just in case every element of $\emptyset$ is also an element of C, i.e. $\forall\, x \in \emptyset\ .\ x \in \mathrm{C}$. However, the empty set contains no elements, so each of its elements satisfies every possible property, including being an element of C. □

The symbol "□" indicates that we have reached the end of the proof.

This proof demonstrates some common patterns that are worth pointing out. First, if we're trying to prove a theorem of the form "Every *blah* has the property *blarg*", it's a good idea to begin by postulating a *blah*, but make no other assumptions about it. This ensures that our *blah* is *generic*, so that whatever is true about this *blah* is true of any *blah* whatsoever. After postulating an arbitrary set C, we expanded the definition of what it means for one set to be a subset of another. This meant *specializing* the definition so that our C played the role of the A in the definition and our $\emptyset$ played the role of B. Finally, we appealed to the fact that the empty set is empty, so that all of its elements – all zero of them – are elements of C. This is called a vacuous quantification. We will learn about more proof techniques as we progress through the course.

Once we have some sets we can use them to describe other sets. The *intersection* of two sets is the set whose elements are contained in *both* of them:

$$\mathrm{A} \cap \mathrm{B} \quad := \quad \{x \mid x \in \mathrm{A} \text{ and } x \in \mathrm{B}\}.$$

In set notation you can read the vertical bar, "|" as "such that". Two sets are called *disjoint* if their intersection is the empty set.

The *union* of two sets is the set whose elements are contained in *at least one* of them:

$$\mathrm{A} \cup \mathrm{B} \quad := \quad \{x \mid x \in \mathrm{A} \text{ or } x \in \mathrm{B}\}.$$

The *relative complement* of B in A contains those elements of A that are *not* elements of B:

$$\mathrm{A} - \mathrm{B} \quad := \quad \{x \mid x \in \mathrm{A} \text{ but } x \notin \mathrm{B}\}.$$

Given any two sets we can form the set of their *ordered pairs*. This is known as the *cartesian product*:

$$\mathrm{A} \times \mathrm{B} \quad := \quad \{(x\,, y) \mid x \in \mathrm{A} \text{ and } y \in \mathrm{B}\}.$$

For example, we have (Ringo , false) $\in$ Beatles $\times$ $\mathbb{2}$. We use this parentheses notation for ordered pairs. The cardinality of the cartesian product of finite sets is the multiplicative product of their respective cardinalities, $|\mathrm{A} \times \mathrm{B}| = |\mathrm{A}| \times |\mathrm{B}|$.

Sometimes we want to collect together the elements of two sets while keeping track of which set each element came from. We can do this using the *disjoint union* (synonym: "tagged union")

$$\mathrm{A} + \mathrm{B} \quad := \quad (\{0\} \times \mathrm{A}) \cup (\{1\} \times \mathrm{B}).$$

The idea is that the first element of each ordered pair, the number 0 or 1, acts as a *tag* to keep track of the source set. The cardinality of the disjoint union of finite sets is the sum of their respective cardinalities: $|\mathrm{A} + \mathrm{B}| = |\mathrm{A}| + |\mathrm{B}|$.

It's worth pointing out that that the definitions of cartesian product and disjoint union above are a little bit arbitrary, in the sense that we could have just as well used something different like

$$\mathrm{A} \times' \mathrm{B} \quad := \quad \{(y\,, x) \mid x \in \mathrm{A} \text{ and } y \in \mathrm{B}\}. \quad \text{and} \quad \mathrm{A} +' \mathrm{B} \quad := \quad (\{\text{false}\} \times \mathrm{A}) \cup (\{\text{true}\} \times \mathrm{B}).$$

These are called *encodings*, and play the same role as binary encodings of data on a computer. Ultimately, it's not important which encoding we use, just that there is at least one such.

Given a set A we can form the set of all of its subsets. This is called the *powerset* of A:

$$\wp(\mathrm{A}) \quad := \quad \{\mathrm{B} \mid \mathrm{B} \subseteq \mathrm{A}\}.$$

For example, $\{\text{George}, \text{Paul}\} \in \wp(\text{Beatles})$ precisely because $\{\text{George}, \text{Paul}\} \subseteq \text{Beatles}$. Sometimes the powerset is written with the notation "2^{A}" because for a finite set $|\wp(\mathrm{A})| = 2^{|\mathrm{A}|}$.

It will be useful for us to be able to reason about lists of things. There are several ways we could encode lists using the other ingredients we discuss this week, but for the sake of expediency we will take the notion of *sequence* as primitive. We distinguish sequences notationally by writing them between square brackets.

Given a set A the set of all finite sequences of elements of A is written "A^*". This operation of taking all finite sequences over a set is sometimes called the *Kleene closure*. For example, $[\text{Paul}, \text{Ringo}, \text{John}, \text{Ringo}] \in \text{Beatles}^*$. A sequence differs from a set in that the order and multiplicity of its elements are both relevant. We overload the cardinality notation $|-|$ to refer to the *length* of a sequence.

2 Functions

Once we have sets at our disposal we want to be able to act on them. One way to do this is with functions. The mathematical notion of function differs somewhat from the one common in programming. The idea is that a function sends elements of one set to elements of another set. So first we should specify the sets involved. A *boundary specification* (synonym: "signature") for a function does precisely this. We write "$f : \mathrm{A} \to \mathrm{B}$" to mean that the function f sends elements of the set A to elements of the set B. The set at the tail of the arrow is called the function's *domain* and the one at its tip is its *codomain* (synonym: "range").

Once we know the boundary of a function we need to explain which element of the codomain each element of the domain gets sent to. We do this using an *element specification*. If the domain is small, we can simply give an enumeration:

$$f : \text{Beatles} \to \mathbb{2} \quad := \quad [\text{John} \mapsto \text{true}, \text{Paul} \mapsto \text{true}, \text{George} \mapsto \text{true}, \text{Ringo} \mapsto \text{false}]$$

The notation "$f\,(x \mapsto y)$", or just "$x \mapsto y$" if f is understood, means that the function f sends the element x of its domain to the element y of its codomain. We can also write this as "$f(x) = y$". Often, we give an element specification in the form of an expression or algorithm:

$$f : \mathbb{N} \to \mathbb{N} \quad := \quad x \mapsto x^2 \quad (\text{or } f(x) = x^2)$$

Note that a function sends each element of its domain set to *exactly one* element of its codomain set.

Two functions are *equal* just in case they have the same boundary sets and agree on where to send each element of the domain. Notice that by this definition *quicksort* and *mergesort* are equal functions, even though they are implemented by different algorithms.

Note also the asymmetry of the roles of the domain and codomain in the specification of a function: we can always trace an element of a function's domain *forward* to wind up at exactly one element of its codomain. But if we try to trace an element of its codomain *backward* we may wind up at either more than or fewer than one element of its domain. Despite this, it is possible to define the "backward version" of a function $f : \mathrm{A} \to \mathrm{B}$. As expected, its domain is B, but in order to account for the possibility of more-than-or-fewer-than one value, we make its codomain $\wp(\mathrm{A})$. Then $f^{-1} : \mathrm{B} \to \wp(\mathrm{A})$ is defined by $y \mapsto \{x \in \mathrm{A} \mid f(x) = y\}$.

In general, a function $f : \mathrm{A} \to \mathrm{B}$ may or may not be either,

injective: to each element of the codomain there corresponds *at most one* element of the domain:
$\forall\, x, y \in \mathrm{A}$. if $f(x) = f(y)$ then $x = y$, or

surjective: to each element of the codomain there corresponds *at least one* element of the domain:
$\forall\, y \in \mathrm{B} \,.\, \exists\, x \in \mathrm{A} \,.\, f(x) = y$.

The "$\exists$" symbol means "there exists" or "for some". Intuitively, an injective function doesn't "collapse" any elements of its domain and a surjective function doesn't "miss" any elements of its codomain. If a function is both injective and surjective then it is,

bijective: to each element of the codomain there corresponds *exactly one* element of the domain:
$\forall\, y \in \mathrm{B} \,.\, \exists!\, x \in \mathrm{A} \,.\, f(x) = y$.

The "$\exists!$" symbol means "there exists a unique" or "for exactly one".

To each set A there is associated a canonical bijective function from A to itself known as the *identity function*. It is written as $\mathrm{id}\ (\mathrm{A}) : \mathrm{A} \to \mathrm{A}$ and defined by $x \mapsto x$.

If we have two functions with *consecutive* boundaries, $f : \mathrm{A} \to \mathrm{B}$ and $g : \mathrm{B} \to \mathrm{C}$, then we can form their *composite function* $f \cdot g : \mathrm{A} \to \mathrm{C}$ (synonym: "$g \circ f$") defined by $x \mapsto g(f(x))$.

A two-argument function is just a function whose domain is a cartesian product, $f : \mathrm{A} \times \mathrm{B} \to \mathrm{C}$. When all of the sets involved are the same set, e.g. $f : \mathrm{A} \times \mathrm{A} \to \mathrm{A}$, we call such a function a *binary operation* on that set. Binary operations on sets are often written using infix notation. Familiar examples include addition and bounded subtraction (where we underflow to 0) on the natural numbers, $\_ + \_\,,\ \_ - \_ : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$. Some binary operations, including addition but not subtraction, are *associative* in the sense that

$$\forall\, x, y, z \in \mathbb{N} \,.\, (x + y) + z = x + (y + z).$$

Since it doesn't matter where we put the parentheses we can just leave them away and write "$x + y + z$" without ambiguity. An associative binary operation on a set is known as a *semigroup*.

Sometimes when we have a binary operation on a set there is an element of the set that is *neutral* for it, in the sense that whenever we apply the operation with the neutral element and any other element, in either order, the result is just the other element. For example, 0 is neutral for $\_ + \_ : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ because

$$\forall\, n \in \mathbb{N} \,.\, 0 + n = n \text{ and } n + 0 = n.$$

A semigroup with a neutral element is called a *monoid*.

We can think of monoid structure as the extension of a binary operation to accommodate any finite number of arguments. On zero arguments it returns the neutral element, on one argument it acts as the identity function, on two arguments it just applies the given binary operation, and on three or more arguments it iterates this operation, bracketed in any way you like thanks to associativity. Monoids abound in mathematics and computer science. For example, the boolean operations of conjunction (and) and disjunction (or) are monoids, as are the set operations of intersection and union, and the string operation of concatenation.

Theorem 2
A binary operation on a set can have at most one neutral element.

Proof. Let $\_ * \_ : \mathrm{A} \times \mathrm{A} \to \mathrm{A}$ be a an arbitrary binary operation. If $\_ * \_$ has no neutral elements (for example because A is empty) then we are done, so suppose that $\_ * \_$ does have neutral elements. We will show that if x and y are both neutral for $\_ * \_$ then $x = y$.

Consider the expression $x * y$. Because x is neutral for $\_ * \_$ it must be that $x * y = y$. But because y is neutral for $\_ * \_$ it must be that $x * y = x$.

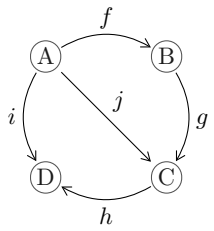
Functions send each element of their domain to exactly one element of their codomain, so it must be that $x = y$. □

Here, we again applied the proof technique of generic instantiation by postulating an arbitrary binary operation on an arbitrary set. Then we used a technique to show that a thing with a certain property is *unique*, that is, there can be at most one of them. We did this by postulating two such things, then using their distinctive property to show that they must have been the same thing all along. I call this technique, "proof by Fight Club". If you've read the novel or watched the film then you can probably guess why.

3 Graphs

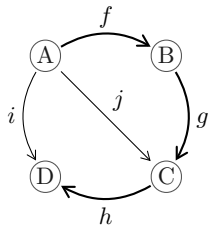
A graph is a structure useful for describing relationships that hold between multiple things. The kind of graph we will use most is sometimes called a "directed multigraph with loops", but we will just call it a "graph". Formally, a *graph* G consists of a set of *vertices* (synonym: "nodes") V, a set of *edges* E, and two boundary functions $s, t : \mathrm{E} \to \mathrm{V}$. The names of these functions are mnemonic for "source" and "target". For an edge e, if $s(e) = \mathrm{A}$ and $t(e) = \mathrm{B}$ then we say that e is an edge from A to B and write "$e : \mathrm{G}\,(\mathrm{A} \to \mathrm{B})$", or just "$e : \mathrm{A} \to \mathrm{B}$", if G is understood. This overloading of the boundary notation shouldn't cause confusion as long as we know whether we're talking about a function between sets or and edge between vertices of a graph.

It is often convenient to describe graphs using diagrams where each vertex is depicted as a labeled dot, circle, etc., and each edge is depicted as a labeled directed arc.



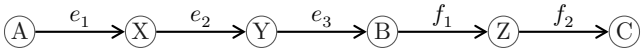
An ordered pair of edges in a graph is *consecutive* if the target of the first is the source of the second. A *path* in a graph is given by an ordered pair of boundary vertices (A , B), together with a finite sequence of consecutive edges es with the given boundary. That is, if es is empty then then $\mathrm{A} = \mathrm{B}$ (this is called a *stationary path*) and otherwise the source of its first edge is A and the target of its last edge is B.

In a diagram representing a graph, a path can be represented as a highlighted sequence of tip-to-tail arcs.



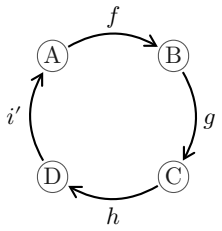
In this graph the highlighted sequence of consecutive edges $[f, g, h]$ is a path from A to D. We can again overload the boundary notation and write $[f, g, h] : \mathrm{A} \to \mathrm{D}$. Note that such a graphical representation becomes ambiguous if the path passes through any vertex more than once. In this case we should write out the sequence of consecutive edges explicitly.

Given consecutive paths $es : \mathrm{A} \to \mathrm{B}$ and $fs : \mathrm{B} \to \mathrm{C}$, the *concatenated path* $es +\!\!+ fs : \mathrm{A} \to \mathrm{C}$ is given by the concatenation of the edge sequences.



Two edges or paths are *parallel* if they have the same boundary as each other. Of course, every edge or path is parallel to itself, but when we say that a graph has parallel edges or paths, we mean excluding self-parallels.

A *cycle* in a graph is a non-stationary path whose source is equal to its target. The following diagram could depict many cycles, including $[f, g, h, i'] : \mathrm{A} \to \mathrm{A}$ and $[h, i', f, g, h, i', f, g] : \mathrm{C} \to \mathrm{C}$.



A *bipartite graph* is one whose vertex set is the disjoint union of two sets ($\mathrm{V} = \mathrm{V}^- + \mathrm{V}^+$), and each edge goes from a vertex in the first set to a vertex in the second set ($\forall\, e \in \mathrm{E}\ .\ s(e) \in \mathrm{V}^-$ and $t(e) \in \mathrm{V}^+$).

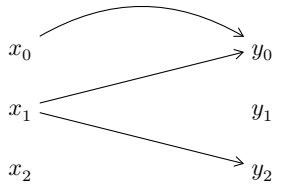
4 Relations

A *predicate* represents a property that may or may not hold of any particular element of a set. For example, evenness is a predicate on the natural numbers that holds of 2, but not of 3. Each predicate is equivalent to a certain subset in the following sense. Given a predicate on a set we can define the subset for which that predicate holds. Going the other way, given a subset of a set we can define the predicate of being an element of that subset.

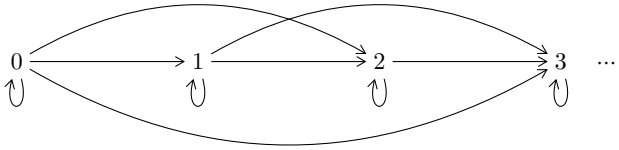
A predicate on a cartesian product of sets is called a (binary) *relation*. For example, $\_ \leq \_$ is a relation on (or "between") $\mathbb{N}$ and $\mathbb{N}$ that holds of $(1\ ,\ 2)$ but not of $(3\ ,\ 2)$. We write "$\mathrm{R} : \mathrm{A} \nrightarrow \mathrm{B}$" to indicate that R is a relation on the sets A and B. We say that A is its *domain* and B its *codomain* just like for functions, but use a different style of arrow to make clear that R is a relation.

For $x \in \mathrm{A}$ and $y \in \mathrm{B}$ we write "$\mathrm{R}\,(x \mathrel{-\!\!\!\rightarrow} y)$", or just "$x \mathrel{-\!\!\!\rightarrow} y$" when R is understood, to indicate that R relates x and y (another common notation is "$x\mathrm{R}y$"). Note that the notation we use for functions, "$\mathrm{R}(x) = y$" is not suitable for relations because R could relate x to many things, or to none.

We can use a diagram for a bipartite graph to represent the element specification of a relation, drawing the elements of its domain set on the left and those of its codomain set on the right, and using directed arcs to join the elements of the two sets that are related.



When the domain and codomain sets of a relation are the same we call it a relation *on* that set. For example, $\_ \leq \_ : \mathbb{N} \nrightarrow \mathbb{N}$ is a relation on the natural numbers. We can represent such a relation as a graph with no parallel edges.



A relation on a set $\mathrm{R} : \mathrm{A} \nrightarrow \mathrm{A}$ is called,

reflexive if it relates each element to itself:

$$\forall\, x \in \mathrm{A}\ .\ \mathrm{R}\,(x \mathrel{-\!\!\!\rightarrow} x)$$

transitive if we can compose paths of relatedness:

$$\forall\, x, y, z \in \mathrm{A}\ .\ \text{if}\ \mathrm{R}\,(x \dashrightarrow y)\ \text{and}\ \mathrm{R}\,(y \dashrightarrow z)\ \text{then}\ \mathrm{R}\,(x \dashrightarrow z)$$

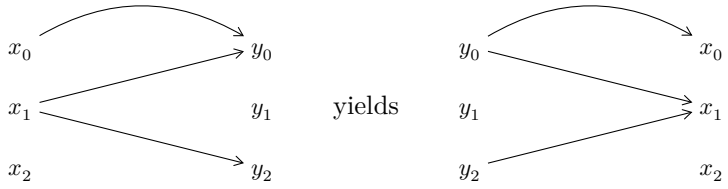
symmetric if relatedness goes both ways:

$$\forall\, x, y \in \mathrm{A}\ .\ \text{if}\ \mathrm{R}\,(x \dashrightarrow y)\ \text{then}\ \mathrm{R}\,(y \dashrightarrow x)$$

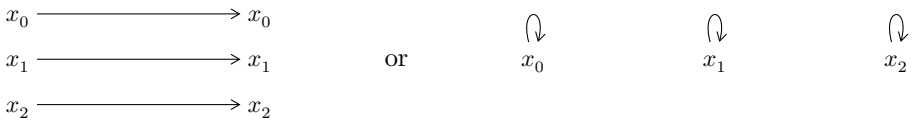
In a diagram for a reflexive relation we can omit loop arcs, which are implied. Similarly, in a diagram for a transitive relation we can omit composite arcs, and for a symmetric relation we can omit the arc arrowheads.

A relation on a set that is both reflexive and transitive is called a *preorder*, and a preorder that is also symmetric is an *equivalence*.

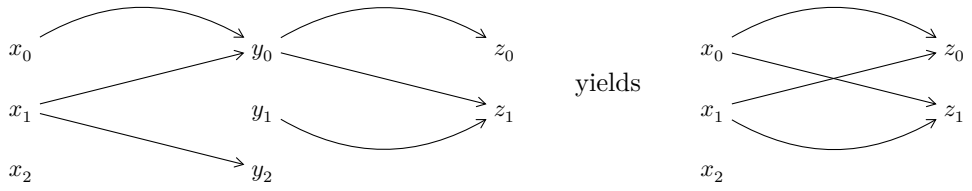
Given a relation $\mathrm{R} : \mathrm{A} \nrightarrow \mathrm{B}$ we can define its *converse relation* $\mathrm{R}^\circ : \mathrm{B} \nrightarrow \mathrm{A}$ by $\mathrm{R}^\circ\,(y \dashrightarrow x)$ just in case $\mathrm{R}\,(x \dashrightarrow y)$. The element diagram for R° looks like the mirror image of that for R.



To each set A there is associated a canonical equivalence relation on A known as the *identity relation*. It is written as $\mathrm{U}(\mathrm{A}) : \mathrm{A} \nrightarrow \mathrm{A}$ and relates each element only to itself.



If we have two relations with *consecutive* boundaries, $\mathrm{R} : \mathrm{A} \nrightarrow \mathrm{B}$ and $\mathrm{S} : \mathrm{B} \nrightarrow \mathrm{C}$, then we can form their *composite relation* $\mathrm{R} \odot \mathrm{S} : \mathrm{A} \nrightarrow \mathrm{C}$ defined by $\mathrm{R} \odot \mathrm{S}\,(x \dashrightarrow z)$ if $\exists\, y \in \mathrm{B}\ .\ \mathrm{R}\,(x \dashrightarrow y)$ and $\mathrm{S}\,(y \dashrightarrow z)$. That is, the composite relation $\mathrm{R} \odot \mathrm{S}$ relates $x \in \mathrm{A}$ with $z \in \mathrm{C}$ just in case there is a path from x to z in the graph we get by combining the graph of R with the graph of S along the elements of B.



Using the concept of relation we can recover that of function by picking out those relations that relate each element of their domain to exactly one element of their codomain. Interestingly, we can also go the other way and define relations in terms of functions. We do this using the powerset trick that we used to define the "backward function" f^{-1} as follows: for a relation $\mathrm{R} : \mathrm{A} \nrightarrow \mathrm{B}$ we define the function $f : \mathrm{A} \to \wp(\mathrm{B})$ by $x \mapsto \{y \in \mathrm{B} \mid \mathrm{R}\,(x \dashrightarrow y)\}$. This way $y \in f(x)$ just in case $\mathrm{R}\,(x \dashrightarrow y)$.