Bourbaki

Contents

1 Sets

1.1 Why

We speak of a collection of objects, pre-specified or possessing a similar property.

1.2 Definition

A **set** is a collection of objects. We use **object** as usual in the English language. So a set is an object with the property that it contains other objects.

In thinking of a set, then, we regularly consider the objects it contains. We call the objects contained in a set the **members** or **elements** of the set. So we say that an object contained in a set is a **member of** or an **element of** the set.

1.2.1 Notation

We denote sets by upper case latin letters: for example, A, B, and C. We denote elements of sets by lower case latin letters: for example, a, b, and c. We denote that an object a is an element of a set A by $a \in A$. We read the notation $a \in A$ aloud as "a

in A." The $\in$ is a stylized ϵ, a mnemonic for "element of". We write $a \notin A$, read aloud as "a not in A," if a is not an element of A.

If we can write down the elements of A, we do so using brace notation. For example, if the set A is such that it contains only the elements a, b, c, we denote A by $\{a, b, c\}$. If the elements of a set are well-known, then we introduce the set in English and name it; often we select the name mnemonically. For example, let L be the set of latin letters.

If the elements of a set satisfy some common condition, then we use the braces and include the condition. For example, let V be the set of vowels. We can denote V by $\{l \in L \mid l \text{ is a vowel}\}$. We read the symbol $\mid$ aloud as "such that." We read the whole notation aloud as "l in L such that l is a vowel." We call the notation **set-builder notation**. Set-builder notation is indispensable for sets defined implicitly by some condition. Here we could have alternatively denoted V by $\{\text{“}a\text{”}, \text{“}e\text{”}, \text{“}i\text{”}, \text{“}o\text{”}, \text{“}u\text{”}\}$. We prefer the former, slighly more concise notation.

1.3 Two Sets

Let A a set. A **subset** of A is a set whose elements are also contained in A. A **superset** if A is a set which contains all the elements of A. Two sets are **equal** if they contain the same elements; equivalently if they are each subsets of each other.

The **power set** of A is the set of subsets of A. The **empty set** is the set containing no elements. The empty set is subset

of every set.

1.3.1 Notation

Let A and B be sets. We denote that A is a subset of B by $A \subset B$. We read the notation $A \subset B$ aloud as "A subset B". We denote that A is equal to B by $A = B$. We read the notation $A = B$ aloud as "A equals B". We denote the empty set by $\varnothing$, read aloud as "empty." We denote the power set of A by 2^A, read aloud as "two to the A."

2 Ordered Pairs

2.1 Why

We speak of objects composed of elements from different sets.

2.2 Definition

Let A and B be non-empty sets. Let $a \in A$ and $b \in B$. An **ordered pair** is the set $\{\{a\}, \{a, b\}\}$. The **cartesian product** of A and B is the set of all ordered pairs.

We observe that two pairs are equal if they have equal elements in the same order. The ordered causes If $A \neq B$, the ordering causes the cartesian product of A and B to differ from the cartesian product of B with A. If $A = B$, however, the symmetry holds.

2.2.1 Notation

We denote the ordered pair $\{\{a\}, \{a, b\}\}$ by (a, b). We denote the cartesian product of A with B by $A \times B$, read aloud as "A cross B." In this notation, if $A \neq B$, then $A \times B \neq B \times A$.

3 Relations

3.1 Why

We want to relate elements of two sets.

3.2 Definition

A **relation** between two non-empty sets A and B is a subset of $A \times B$. A relation on a single set C is a subset of $C \times C$.

3.2.1 Notation

We denote relations with upper case capital latin letters because they are sets. Let R be a relation on A and B. We denote that $(a, b) \in R$ by aRb, read aloud as "a in relation R to b."

Often, instead of latin letters we use other symbols. For example, $\sim$, $=$, $<$, $\leq$, $\prec$, and $\preceq$.

3.3 Properties

Let R be a relation on a non-empty set A. R is **reflexive** if

$$(a, a) \in R$$

for all $a \in A$. R is **transitive** if

$$(a, b) \in R \wedge (b, c) \in R \implies (a, c) \in R$$

for all $a, b, c \in A$. R is **symmetric** if

$$(a, b) \in R \implies (b, a) \in R$$

for all $a, b \in A$. R is **anti-symmetric** if

$$(a, b) \in R \implies (b, a) \notin R$$

for all $a, b \in A$.

4 Functions

4.1 Why

We want a notion for a correspondence between two sets.

4.2 Definition

A **functional** relation on two sets relates each element of the first set with a unique element of the second set. A **function** is a functional relation.

The **domain** of the function is the first set and **codomain** of the function is the second set. The function **maps** elements **from** the domain **to** the codomain. We call the codomain element associated with the domain element the **result** of **applying** the function to the domain element.

4.2.1 Notation

Let A and B be sets. If A is the domain and B the codomain, we denote the set of functions from A to B by $A \to B$, read aloud as "A to B".

A function is an element of th set $A \to B$, so we denote them

by lower case latin letters, especially f, g, and h. Of course, f is a mnemonic for function; g and h follow f in the alphabet. We denote that $f \in A \to B$, by $f : A \to B$, read aloud as "f from A to B".

Let $f : A \to B$. For each element $a \in A$, we denote the result of applying f to a by $f(a)$, read aloud "f of a." We sometimes drop the parentheses, and write the result as f_a, read aloud as "f sub a."

Let $g : A \times B \to C$. We often write $g(a, b)$ or g_{ab} instead of $g((a, b))$. We read $g(a, b)$ aloud as "g of a and b". We read g_{ab} aloud as "g sub a b."

4.3 Properties

Let $f : A \to B$. The **image** of a set $C \subset A$ is the set $\{f(c) \in B \mid c \in C\}$. The **range** of f is the image of the domain. The **inverse image** of a set $D \subset B$ is the set $\{a \in A \mid f(a) \in B\}$.

The range need not equal the codomain; though it, like every other image, is a subset of the codomain. The function maps to domain **on** to the codomain if the range and codomain are equal; in this case we call the function **onto**. This language suggests that every element of the codomain is used by f. It means that for each element b of the codomain, we can find an element a of the domain so that $f(a) = b$.

An element of the codomain may be the result of several elements of the domain. This overlapping, using an element of the

codomain more than once, is a regular occurrence. If a function is a unique correspondence in that every domain element has a different result, we call it **one-to-one**. This language is meant to suggest that each element of the domain corresponds to one and exactly one element of the codomain, and vice versa.

4.3.1 Notation

Let $f : A \to B$. We denote the image of $C \subset A$ by $f(C)$, read aloud as "f of C." This notation is overloaded: for $c \in C$, $f(c) \in A$, whereas $f(C) \subset A$. Read aloud, the two are indistinguishable, so we must be careful to specify whether we mean an element c or a set C. The property that f is onto can be written succintly as $f(A) = B$. We denote the inverse image of $D \subset B$ by $f^{-1}(D)$, read aloud as "f inverse D."

5 Natural Numbers

5.1 Why

We want to count, forever.

5.2 Definition

We define the set of **natural numbers** implicitly. There is an element of the set which we call **one**. Then we say that for each element n of the set, there is a unique corresponding element called the **successor** of n which is also in the set. The **successor function** is the implicitly defined a function from the set into itself associating elements with their successors. We call the elements **numbers** and the refer to the set itself as the **naturals**.

To recap, we start by knowing that one is in the set, and the successor of one is in the set. We call the successor of one **two**. We call the successor of two **three**. And so on using the English language in the usual manner. We are saying, in the language of sets, that the essence of counting is starting with one and adding one repeatedly.

5.2.1 Notation

We denote the set of natural numbers by N, a mnemonic for natural. We often denote elements of N by n, a mnemonic for number, or m, a letter close to n. We denote the element called one by 1.

5.3 Induction

We assert two additonal self-evident and indispensable properties of these natural numbers. First, one is the successor of no other element. Second, if we have a subset of the naturals containing one with the property that it contains successors of its elements, then that set is equal to the natural numbers. We call this second property the **principle of mathematical induction.**

These two properties, along with the existence and uniqueness of successors are together called **Peano's axioms** for the natural numbers. When in familiar company, we freely assume Peano's axioms.

5.4 Notation

As an exercise in the notation assumed so far, we can write Peano's axioms: N is a set along with a function $s : N \to N$ such that

1. $s(n)$ is the successor of n for all $n \in N$.

2. s is one-to-one; $s(n) = s(m) \implies m = n$ for all $m, n \in N$.

3. There does not exist $n \in N$ such that $s(n) = 1$.

4. If $T \subset N$, $1 \in T$, and $s(n) \in T$ for all $n \in T$, then $T = N$.

6 Graphs

6.1 Why

We want to visualize relations.

6.2 Definition

A **graph** is a set and a relation on the set. The graph is **undirected** if the relation is symmetric; otherwise the graph is **directed**.

A **vertex** of the graph is an element of the set. The set is called the **vertex set**. An **edge** of the graph is an element of the relation. The relation is called the **edge set**.

6.2.1 Notation

We denote the vertex set by V, a mnemonic for vertex. We denote the edge set by E, a mnemonic for edge. We denote a graph by (V, E). If the vertex set is assumed we can unambiguously refer to the graph by E.

6.2.2 Visualization

We visualize the graph by drawing a point for each vertex. If two vertices u and v are in relation, we draw a line from the point corresponding to u to the point corresponding to v with an arrow at the point corresponding to v. If the graph is undirected, we omit arrows.

6.3 Paths

A path in a relation is a sequence of elements in which consecutive elements are related. A path **cycles** if an element appears more than once. A path is **finite** if the sequence is finite. A finite path is a **loop** if it cycles once.

7 Trees

7.1 Why

Tree branches split and do not recombine. We formalize this property in the language of graphs.

7.2 Definition

A **tree** is a connected acyclic graph.

7.2.1 Notation

Let (V, E) be a tree. When the vertex set is clear from context, we use use T, a mnemonic for "tree," to denote the edge set. We denote the set of trees on the vertex set V by $T(V)$.

7.3 Properties

Proposition 1. *In any tree, there is only one path between any two vertices.*

8 Graph Cliques

8.1 Why

We speak of the complete subgraphs of a graph.

8.2 Definition

A **complete** graph is one for which an edge exists between any two nodes.

A **subgraph** of a given graph is a graph whose vertex set is a subset of the given vertex set and whose edge set is the subset of given edges connecting vertices in the vertex subset. With reference to the underling graph, then, a subgraph can be specified completely by its vertex set.

A **clique** of a given graph is a complete subgraph of that graph. When speaking of the cliques of a given graph, we identify the cliques with their vertex set. The relation contained in gives a partial order on cliques. A clique is **maximal** if it maximal with respect to this relation; i.e., it is contained in no other clique. As a convention, we include $\varnothing$ as a clique.

8.2.1 Notation

Let (V, E) a graph. We denote a clique by $C \subset V$, a mnemonic for clique.

9 Function Composition

9.1 Why

We want a notion for applying two functions one after the other. We apply a first function then a second function.

9.2 Definition

Consider two functions for which the codomain of the first function is the domain of the second function.

The **composition** of the second function with the first function is the function which associates each element in the first's domain with the element in the second's codomain that the second function associates with the result of the first function.

The idea is that we take an element in the first domain. We apply the first function to it. We obtain an element in the first's codomain. This result is an element of the second's domain. We apply the second function to this result. We obtain an element in the second's codomain. The composition of the second function with the first is the function so constructed.

9.2.1 Notation

Let A, B, C be non-empty sets. Let $f : A \to B$ and $g : B \to C$. We denote the composition of g with f by $g \circ f$ read aloud as "g composed with f." To make clear the domain and comdomain, we denote the composition $g \circ f : A \to C$.

In previously introduced notation, $g \circ f$ satisfies

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

for all $a \in A$.

9.3 Inverses

The **identity function** on a set is a function which associates each element with itself. A function

A second function is an **inverse function** of a first function if the domain of the first function is the codomain of the second function, the domain of the second function is the codomain of the first function, and the composition of the first function with the second is the identity function on the first domain and the composition of the second function on the first is the identity function on the second domain.

10 Function Inverses

10.1 Why

We want a notion of reversing functions.

10.2 Definition

An **identity function** is a relation on a set which is functional and reflexive. It associates each element in the set with itself. There is only one identity function associated to each set.

Consider two functions for which the codomain of the first function is the domain of the second function and the codomain of the second function is the domain of the first function. These functions are **inverse functions** if the composition of the second with the first is the identity function on the first's domain and the composition of the first with the second is the identity function on the second's domain.

In this case we say that the second function is an **inverse** of the second, and vice versa. When an inverse exists, it is unique, so we refer to the **inverse** of a function.

10.2.1 Notation

Let A a non-empty set. We denote the identity function on A by id_A, read aloud as "identity on A." id_A maps A onto A.

Let A, B be non-empty sets. Let $f : A \to B$ and $g : B \to A$ be functions. f and g are inverse functions if $g \circ f = \mathrm{id}_A$ and $f \circ g = \mathrm{id}_B$.

10.3 The Inverse

We discuss existence and uniqueness of an inverse.

Proposition 2. *Let $f : A \to B$, $g : B \to A$, and $h : B \to A$.*

If g and h are both inverse functions of f, then $g = h$.

Proof. □

Proposition 3. *If a function is one-to-one and onto, it has an inverse.*

Proof. □

11 Order Relations

11.1 Why

We want to handle elements of a set in a particular order.

11.2 Definition

Let R be a relation on a non-empty set A. R is a **partial order** if it is reflexive, transitive, and anti-symmetric.

A **partially ordered set** is a set and a partial order. The language partial is meant to suggest that two elements need not be comparable. For example, suppose R is $\{(a, a) \mid a \in A\}$; we may justifiably call this no order at all and call A totally unordered, but it is a partial order by our definition.

Often we want all elements of the set A to be comparable. We call R **connexive** if for all $a, b \in A$, $(a, b) \in R$ or $(b, a) \in R$. If R is a partial order and connexive, we call it a **total order**.

A **totally ordered set** is a set together with a total order. The language is a faithful guide: we can compare any two elements. Still, we prefer one word to three, and so we will use the shorter term **chain** for a totally ordered set; other terms include **simply ordered set** and **linearly ordered set**.

11.2.1 Notation

We denote total and partial orders on a set A by $\preceq$. We read $\preceq$ aloud as "precedes or equal to" and so read $a \preceq b$ aloud as "a precedes or is equal to b." If $a \preceq b$ but $a \neq b$, we write $a \prec b$, read aloud as "a precedes b."

12 Algebra

12.1 Why

We want to combine set elements to get other set elements.

12.2 Basics

An **operation** on a set is a function from ordered pairs of elements in the set to the same set. We use operations to combine the elements. We operate on pairs. An **algebra** is a set and an operation. We call the set the **ground set**.

12.2.1 Notation

Let A a set and $g : A \times A \to A$. We commonly forego the notation $g(a, b)$ and instead write $a\, g\, b$. We call this style **infix notation**.

Using lower case latin letters for every the elements and for the operation is confusing, but we often have special symbols for particular operations. Examples of such symbols include $+$, $-$, $\cdot$, $\circ$, and $\star$.

If we had a set A and an operation $+ : A \times A \to A$, we would

write $a+b$ for the result of applying $+$ to (a, b). In denoting the algebra, we would say let $(A, +)$ be an algebra.

12.3 Operation Properties

An operation **commutes** if the result of two elements is the same regardless of their order; we call the operation **commutative**

An operation **associates** if given any three elements in order it doesn't matter whether we first operate on the first two and then with the result of the first two the third, or the second two and with the result of the second two the first.

A first operation over a set **distributes** over a second operation over the same set if the result of applying the first operation to an element and a result of the second operation is the same as applying the second operation to the results of the first operation with the arguments of the second operation.

12.3.1 Notation

Let $(A, +)$ an algebra.

We denote that $+$ commutes by asserting

$$a + b = b + a$$

for all $a, b \in A$. We denote that $+$ associates by asserting

$$(a + b) + c = (a + b) + c$$

for all $a, b, c \in A$. Let $(A, \cdot)$ a second algebra over the same set. We denote that $\cdot$ distributes over $+$ by

$$a \cdot (b + c) = (a \cdot b) + (a \cdot c)$$

for all $a, b, c \in A$.

12.4 Identity Elements

We call $e \in A$ an **identity element** if (1) $e + a = e$ and (2) $a + e = e$ for all $a \in A$. If only (1) holds, we call e a **left identity**. If only (2) holds, we call e a **right identity**.

12.5 Inverse Elements

We call $b \in A$ an **inverse element** of $a \in A$ if (1) $b + a = e$ and (2) $a + b = e$. If only (1) holds, we call e a **left inverse**. If only (2) holds, we call e a **right inverse**.

13 Set Operations

13.1 Why

We want to consider the elements of two sets together at once, and other sets created from two sets.

13.2 Definitions

Let A and B be two sets.

The **union** of A with B is the set whose elements are in either A *or* B *or* both. The key word in the definition is *or*.

The **intersection** of A with B is the set whose elements are in both A *and* B. The keyword in the definition is *and*.

Viewed as operations, both union and intersection commute; this property justifies the language "with." The intersection is a subset of A, of B, and of the union of A with B.

The **symmetric difference** of A and B is the set whose elements are in the union but not in the intersection. The symmetric difference commutes because both union and intersection commute; this property justifies the language "and." The symmetric difference is a subset of the union.

Let C be a set containing A. The **complement** of A in C is the symmetric difference of A and C. Since $A \subset C$, the union is C and the intersection is A. So the complement is the "left-over" elements of B after removing the elements of A.

We call these four operations **set-algebraic operations**.

13.2.1 Notation

Let A, B be sets. We denote the union of A with B by $A \cup B$, read aloud as "A union B." $\cup$ is a stylized U. We denote the intersection of A with B by $A \cap B$, read aloud as "A intersect B." We denote the symmetric difference of A and B by $A \Delta B$, read aloud as "A symdiff B." "Delta" is a mnemonic for difference.

Let C be a set containing A. We denote the complement of A in C by $C - A$, read aloud as "C minus A."

13.2.2 Results

Proposition 4. *For all sets A and B the operations $\cup$, $\cap$, and Δ commute.*

Proposition 5. *Let S a set. For all sets $A, B \subset S$,*

$$(1) \quad S - (A \cup B) = (S - A) \cap (S - B)$$

$$(2) \quad S - (A \cap B) = (S - A) \cup (S - B).$$

Proposition 6. *Let S a set. For all sets $A, B \subset S$,*

$$A \Delta B = (A \cup B) \cap C_S(A \cap B)$$

$\boxed{TODO : notation}$

14 Arithmetic

14.1 Why

Counting one by one is slow so we define an algebra on the naturals.

14.2 Sums and Addition

Let m and n be two natural numbers. If we apply the successor function to m n times we obtain a number. If we apply the successor function to n m times we obtain a number. Indeed, we obtain the same number in both cases. We call this number the **sum** of m and n. We say we **add** m to n, or vice versa. We call this symmetric operation mapping (m, n) to their sum **addition**.

14.2.1 Notation

We denote the operation of addition by $+$ and so denote the sum of the naturals m and n by $m + n$.

14.3 Products and Multiplication

Let m and n naturals. If we add n copies of m we obtain a number. If we add m copies of n we obtain a number. Indeed, we obtain the same number in both cases. We call this number the **product** of m and n. We say we **multiply** m to n, or vice versa. We call this symmetric operation mapping (m, n) to their product **multiplication**.

14.3.1 Notation

We denote the operation of multiplication by $\cdot$ and so denote the product of the naturals m and n by $m \cdot n$.

15 Equivalence Relations

15.1 Why

We want to handle at once all elements which are indistinguishable or equivalent in some aspect.

15.2 Definition

A relation R on a set A is an **equivalence relation** if it is reflexive, symmetric, and transitive.

For an element $a \in A$, we call the set of elements in relation R to a the **equivalence class** of a. The key observation, recorded and proven below, is that the equivalence classes partition the set A. A frequent technique is to define an appropriate equivalence relation on a large set A and then to work with the set of equivalence classes of A.

We call the set of equivalence classes the **quotient set** of A under R. An equally good name is the divided set of A under R, but this terminology is not standard. The language in both cases reminds us that $\sim$ partitions the set A into equivalence classes.

15.2.1 Notation

If R is an equivalence relation on a set A, we use the symbol $\sim$. When alone, $\sim$ is read aloud as "sim," but we still read $a \sim b$ aloud as "a equivalent to b." We denote the quotient set of A under $\sim$ by $A/\sim$, read aloud as "A quotient sim".

15.2.2 Results

$\boxed{TODO}$

16 Families

16.1 Why

We want to generalize operations beyond two objects.

16.2 Definition

Let A, B be non-empty sets. A **family** of elements of a first set **indexed** by elements of a second set is the range of a function from the second set to the first set. We call second set the **index set**.

If the index set is a finite set, we call the family a **finite family**. If the index set a countable set, we call the family a **countable family**. If the index set is an uncountable set, we call the family a **uncountable family**.

If the codomain is a set of sets, we call the family a **family of sets**. We often use a subset of the whole natural numbers as the index set. In this case, and for other indexed sets with orders, we call the family an **ordered family**

16.2.1 Notation

Let A be a non-empty set. We denote the index set by I, a mnemonic for index. For $i \in I$, let we denote the result of applying the function to i by a_i; the notation evokes evokes function notation but avoids naming the function.

We denote the family of a_α indexed with I by $\{a_\alpha\}_{\alpha \in I}$, which is short-hand for set-builder notation. We read this notation "a sub-alpha, alpha in I."

16.3 Operations

The **pairwise extension** of a commutative operation is the function from finite families of the ground set to the ground set obtained by applying the operation pairwise to elements.

The **ordered pairwise extension** of an operation is the function from finite families ground set to the ground set obtained by applying the operation pairwise to elements in order.

16.3.1 Notation

Let $(A, +)$ be an algebra and $\{A_i\}_{i=1}^n$ a finite family of elements of A. We denote the pairwise extension by

$$\overset{n}{\underset{i=1}{+}} A_i$$

16.4 Family Set Algebra

We define the set whose elements are the objects which are contained in at least one family member the **family union**. We define the set whose elements are the objects which are contained in all of the family members the **family intersection**.

16.4.1 Notation

We denote the family union by $\cup_{\alpha \in I} A_\alpha$. We read this notation as "union over alpha in I of A sub-alpha." We denote family intersection by $\cap_{\alpha \in I} A_\alpha$. We read this notation as "intersection over alpha in I of A sub-alpha."

16.4.2 Results

Proposition 7. *For an indexed family* $\{A_\alpha\}_{\alpha \in I}$ *in* S*, if* $I = \{i, j\}$ *then*

$$\cup_{\alpha \in I} A_\alpha = A_i \cup A_j$$

and

$$\cap_{\alpha \in I} A_\alpha = A_i \cap A_j.$$

Proposition 8. *For an indexed family* $\{A_\alpha\}_{\alpha \in I}$ *in* S*, if* $I = \varnothing$*, then*

$$\cup_{\alpha \in I} A_\alpha = \varnothing$$

and

$$\cap_{\alpha \in I} A_\alpha = S.$$

Proposition 9. *For an indexed family* $\{A_\alpha\}_{\alpha \in I}$ *in* S*.*

$$C_S(\cup_{\alpha \in I} A_\alpha) = \cap_{\alpha \in I} C_S(A_\alpha)$$

and

$$C_S(\cap_{\alpha \in I} A_\alpha) = \cup_{\alpha \in I} C_S(A_\alpha).$$

17 Partitions

17.1 Why

We divide a set into disjoint subsets whose union is the whole set. In this way we can handle each subset of the main set individually, and so handle the entire set piece by piece.

17.2 Definition

A **disjoint family** of sets is a family for which the intersection of any two member sets is empty. A **partition** of a set is a disjoint family of subsets of the set whose union is the set. A **piece** of a partition is an element of the family.

17.2.1 Notation

No new notation for partitions. Instead, we record the properties of partitions in previously introduced notation.

Let A be a set and $\{A_\alpha\}_{\alpha \in I}$ a family of subsets of A. We denote the condition that the family is disjoint by $A_\alpha \cap A_\beta = \varnothing$, for all $\alpha, \beta \in I$, We denote the condition that the family union is A by $\cup_{\alpha \in I} A_\alpha = A$.

18 Direct Products

18.1 Why

We can profitably generalize the notion of cartesian product to families of sets indexed by the natural numbers.

18.2 Direct Products

The **direct product** of family indexed by a subset of the naturals is the set whose elements are ordered sequences of elements from each set in the family. The ordering on the sequences comes from the natural ordering on N. If the index set is finite, we call the elements of the direct product **n-tuples**. If the index set is the natural numbers, and every set in the family is the same set A, we call the elements of the direct product the **sequences** in A.

18.2.1 Notation

For a family $\{A_\alpha\}_{\alpha \in I}$ of S with $I = \{1, \ldots, n\}$, we denote the direct product by

$$\prod_{i=1}^{n} A_i.$$

We read this notation as "product over alpha in I of A sub-alpha." We denote an element of $\prod_{i=1}^{n} A_i$ by $(a_1, a_2, \ldots, a_n)$ with the understanding that $a_1 \in A_1, a_2 \in A_2, \ldots, a_n \in A_n$.

If I is the set of natural numbers we denote the direct product by

$$\prod_{i=1}^{\infty} A_i.$$

We denote an element of $\prod_{i=1}^{\infty} A_i$ by (a_i) with the understanding that $a_i \in A_i$ for all $i = 1, 2, 3, \ldots$. If $A_i = A$ for all $i = 1, 2, 3, \ldots$, then (a_i) is a sequence in A.

19 Sequences

19.1 Why

We introduce language for the steps of an infinite process.

19.2 Definition

A **sequence** is a function from the natural numbers to a set.

Equivalently, a sequence is an element of a direct product of a family of sets for which each set in the family is identical and the index set is the natural numbers.

19.2.1 Notation

20 Monotone Sequences

20.1 Why

If the base set of a sequence has a partial order, then we can discuss its relation to the order of sequence.

20.2 Definition

A sequence on a partially ordered set is **non-decreasing** if whenever a first index precedes a second index the element associated with the first index precedes the element associated with the second element. A sequence on a partially ordered set is **increasing** if it is non-decreasing and no two elements are the same. An increasing sequence is non-decreasing.

A sequence on a partially ordered set is **non-increasing** if whenever a first index precedes a second index the element associated with the first index succedes the element associated with the second element. A sequence on a partially ordered set is **decreasing** if it is non-increasing and no two elements are the same. A decreasing sequences is non-increasing.

A sequence on a partially ordered set is **monotone** if it is non-decreasing, or non-increasing. A sequence on a partially or-

dered set is **strictly monotone** if it is decreasing, or increasing.

20.2.1 Notation

Let A a non-empty set with partial order $\preceq$. Let $\{a_n\}_n$ a sequence in A.

The sequence is non-decreasing if $n \leq m \implies a_n \preceq a_m$, and increasing if $n < m \implies a_n \prec a_m$. The sequence is non-increasing if $n \leq m \implies a_n \succeq a_m$, and decreasing if $n < m \implies a_n > a_m$.

20.3 Examples

Example 10. *Let A a non-empty set and $\{A_n\}_n$ a sequence of sets in 2^A. Partially order elements of 2^A by the relation contained in.*

21 Nets

21.1 Why

We generalize the notion of sequence to index sets beyond the naturals.

21.2 Definition

A sequence is a function on the natural numbers; this set has two important properties: (a) we can order the natural numbers and (b) we can always go "further out."

To elaborate on property (b): if handed two natural numbers m and n, we can always find another, for example $\max\{m, n\}+1$, larger than m and n. We might think of larger as "further out" from the first natural number: 1.

Combining these to observations, we define a directed set:

Definition 11. *A **directed set** is a set D with a partial order $\preceq$ satisfying one additional property: for all $a, b \in D$, there exists $c \in D$ such that $a \preceq c$ and $b \preceq c$.*

Definition 12. *A **net** is a function on a directed set.*

A sequence, then, is a net. The directed set is the set of natural numbers and the partial order is $m \preceq n$ if $m \leq n$.

21.2.1 Notation

Directed sets involve a set and a partial order. We commonly assume the partial order, and just denote the set. We use the letter D as a mnemonic for directed.

For nets, we use function notation and generalize sequence notation. We denote the net $x : D \to A$ by $\{a_\alpha\}$, emulating notation for sequences. The use of α rather than n reminds us that D need not be the set of natural numbers.

22 Categories

22.1 Why

We generalize the notion of sets and functions.

22.2 Definition

A **category** is a collection of objects together with a set of **category maps** for each ordered pair of objects. The set of maps has a binary operation called **category composition**, whose induced algebra is associative and contains identities.

As the fundamental example, consider the category whose objects are sets and whose maps are functions. The sets are the objects of the category. The functions are the maps. The rule of composition is ordinary function composition. The map identities are the identity functions. We call this category the **category of sets**.

22.2.1 Notation

Our notation for categories is guided by our generalizing the notions of set and functions.

We denote categories with upper-case latin letters in script; for example, $\mathcal{C}$. We read $\mathcal{C}$ aloud as "script C." Upper case latin letters remind that the category is a set of objects. The script form reminds that these objects may themselves be sets.

We denote the objects of a category by upper-case latin letters, for example A, B, C; an allusion to the idea that these generalize sets. We denote the set of maps for an ordered pair of objects (A, B) by $A \to B$; an allusion to the function notation. We denote members of $A \to B$ using lower case latin letters, for example f, g, h; an allusion to our function notation.

23 Groups

23.1 Why

We generalize the algebraic structure of addition over the integers.

23.2 Definition

A **group** is an algebra with: (1) an associative operation, (2) an identity element, and (3) an inverse for each element. We call the operation of the algebra **group addition**. A **commutative group** is a group whose operation commutes.

23.2.1 Notation

TODO

24 Rings

24.1 Why

We generalize the algebraic structure of addition and multiplication over the integers.

24.2 Definition

A **ring** is two algebras over the same ground set with: (1) the first algebra a commutative group (2) an identity element in the second algebra, and (3) the operation of the second algebra distributes over the operation of the first algebra.

We call the operation of the first algebra **ring addition**. We call the operation of the second algebra **ring multiplication**.

24.2.1 Notation

TODO

25 Fields

25.1 Why

We generalize the algebraic structure of addition and multiplication over the rationals.

25.2 Definition

A **field** is two algebras over the same ground set with: (1) both algebras are commutative groups (2) the operation of the second algebra distributes over the operation of the first algebra.

We call the operation of the first algebra **field addition**. We call the operation of the second algebra **field multiplication**.

25.2.1 Notation

TODO

26 Vectors

26.1 Why

We speak of objects which we can add and scale.

26.2 Definition

26.2.1 Notation

27 Norms

27.1 Why

We want to measure the size of an element in a vector space.

27.2 Definition

A **norm** is a real-valued functional that is (a) non-negative, (b) definite, (c) absolutely homogeneous, (d) and satisifies a triangle inequality. The triangle inequality property requires that the norm applied to the sum of any two vectors is less than the sum of the norms.

27.2.1 Examples

Example 13. *The absolute value function is a norm on the vector space of real numbers.*

Example 14. *The Euclidean distance is a norm on the various real spaces.*