IsarMathLib

Sławomir Kołodyński, Daniel de la Concepción Sáez

February 22, 2021

Abstract

This is the proof document of the IsarMathLib project version 1.15.1. IsarMathLib is a library of formalized mathematics for Isabelle2021 (ZF logic).

Contents

1 Introduction to the IsarMathLib project

theory Introduction imports ZF.equalities

begin

This theory does not contain any formalized mathematics used in other theories, but is an introduction to IsarMathLib project.

1.1 How to read IsarMathLib proofs - a tutorial

Isar (the Isabelle's formal proof language) was designed to be similar to the standard language of mathematics. Any person able to read proofs in a typical mathematical paper should be able to read and understand Isar proofs without having to learn a special proof language. However, Isar is a formal proof language and as such it does contain a couple of constructs whose meaning is hard to guess. In this tutorial we will define a notion and prove an example theorem about that notion, explaining Isar syntax along the way. This tutorial may also serve as a style guide for IsarMathLib contributors. Note that this tutorial aims to help in reading the presentation of the Isar language that is used in IsarMathLib proof document and HTML rendering on the FormalMath.org site, but does not teach how to write proofs that can be verified by Isabelle. This presentation is different than the source processed by Isabelle (the concept that the source and presentation look different should be familiar to any LaTeX user). To learn how to write Isar proofs one needs to study the source of this tutorial as well.

The first thing that mathematicians typically do is to define notions. In Isar this is done with the definition keyword. In our case we define a notion of two sets being disjoint. We will use the infix notation, i.e. the string –is disjoint with″ put between two sets to denote our notion of disjointness. The left side of the ≡ symbol is the notion being defined, the right side says how we define it. In Isabelle/ZF 0 is used to denote both zero (of natural numbers) and the empty set, which is not surprising as those two things are the same in set theory.

definition
  AreDisjoint (infix –is disjoint with″ 90) where
  A –is disjoint with″ B ≡ A ∩ B = 0

We are ready to prove a theorem. Here we show that the relation of being disjoint is symmetric. We start with one of the keywords "theorem", "lemma" or "corollary". In Isar they are synonymous. Then we provide a name for the theorem. In standard mathematics theorems are numbered. In Isar we can do that too, but it is considered better to give theorems meaningful names. After the "shows" keyword we give the statement to show. The ⟷ symbol denotes the equivalence in Isabelle/ZF. Here we want to show that "A is disjoint with B iff and only if B is disjoint with A". To prove this fact we show two implications - the first one that A –is disjoint with″ B implies B –is disjoint with″ A and then the converse one. Each of these implications is formulated as a statement to be proved and then proved in a subproof like a mini-theorem. Each subproof uses a proof block to show the implication. Proof blocks are delimited with curly brackets in Isar. Proof block is one of the constructs that does not exist in informal mathematics, so it may be confusing. When reading a proof containing a proof block I suggest to focus first on what is that we are proving in it. This can be done by looking at the first line or two of the block and then at the last statement. In our case the block starts with "assume A –is disjoint with″ B and the last statement is "then have B –is disjoint with″ A". It is a typical pattern when someone needs to prove an implication: one assumes the antecedent and then shows that the consequent follows from this assumption. Implications are denoted with the ⟶ symbol in Isabelle. After we prove both implications we collect them using the "moreover" construct. The keyword "ultimately" indicates that what follows is the conclusion of the statements collected with "moreover". The "show" keyword is like "have", except that it indicates that we have arrived at the claim of the theorem (or a subproof).

theorem disjointness˙symmetric:
  shows A –is disjoint with″ B ⟷ B –is disjoint with″ A
⟨*proof*⟩

1.2 Overview of the project

The Fol1, ZF1 and Nat˙ZF˙IML theory files contain some background material that is needed for the remaining theories.

Order˙ZF and Order˙ZF˙1a reformulate material from standard Isabelle's Order theory in terms of non-strict (less-or-equal) order relations. Order˙ZF˙1 on the other hand directly continues the Order theory file using strict order relations (less and not equal). This is useful for translating theorems from Metamath.

In NatOrder˙ZF we prove that the usual order on natural numbers is linear.

The func1 theory provides basic facts about functions. func˙ZF continues this development with more advanced topics that relate to algebraic properties of binary operations, like lifting a binary operation to a function space, associative, commutative and distributive operations and properties of functions related to order relations. func˙ZF˙1 is about properties of functions related to order relations.

The standard Isabelle's Finite theory defines the finite powerset of a set as a certain "datatype" (?) with some recursive properties. IsarMathLib's Finite1 and Finite˙ZF˙1 theories develop more facts about this notion. These two theories are obsolete now. They will be gradually replaced by an approach based on set theory rather than tools specific to Isabelle. This approach is presented in Finite˙ZF theory file.

In FinOrd˙ZF we talk about ordered finite sets.

The EquivClass1 theory file is a reformulation of the material in the standard Isabelle's EquivClass theory in the spirit of ZF set theory.

FiniteSeq˙ZF discusses the notion of finite sequences (a.k.a. lists).

InductiveSeq˙ZF provides the definition and properties of (what is known in basic calculus as) sequences defined by induction, i. e. by a formula of the form $a_0 = x,\ a_{n+1} = f(a_n)$.

Fold˙ZF shows how the familiar from functional programming notion of fold can be interpreted in set theory.

Partitions˙ZF is about splitting a set into non-overlapping subsets. This is a common trick in proofs.

Semigroup˙ZF treats the expressions of the form $a_0 \cdot a_1 \cdot .. \cdot a_n$, (i.e. products of finite sequences), where "$\cdot$" is an associative binary operation.

CommutativeSemigroup˙ZF is another take on a similar subject. This time we consider the case when the operation is commutative and the result of depends only on the set of elements we are summing (additively speaking), but not the order.

The Topology˙ZF series covers basics of general topology: interior, closure, boundary, compact sets, separation axioms and continuous functions.

Group˙ZF, Group˙ZF˙1, Group˙ZF˙1b and Group˙ZF˙2 provide basic facts of the group theory. Group˙ZF˙3 considers the notion of almost homomorphisms that is nedeed for the real numbers construction in Real˙ZF.

The TopologicalGroup connects the Topology˙ZF and Group˙ZF series and starts the subject of topological groups with some basic definitions and facts.

In DirectProduct˙ZF we define direct product of groups and show some its basic properties.

The OrderedGroup˙ZF theory treats ordered groups. This is a suprisingly large theory for such relatively obscure topic.

Ring˙ZF defines rings. Ring˙ZF˙1 covers the properties of rings that are specific to the real numbers construction in Real˙ZF.

The OrderedRing˙ZF theory looks at the consequences of adding a linear order to the ring algebraic structure.

Field˙ZF and OrderedField˙ZF contain basic facts about (you guessed it) fields and ordered fields.

Int˙ZF˙IML theory considers the integers as a monoid (multiplication) and an abelian ordered group (addition). In Int˙ZF˙1 we show that integers form a commutative ring. Int˙ZF˙2 contains some facts about slopes (almost homomorphisms on integers) needed for real numbers construction, used in Real˙ZF˙1.

In the IntDiv˙ZF˙IML theory we translate some properties of the integer quotient and reminder functions studied in the standard Isabelle's IntDiv˙ZF theory to the notation used in IsarMathLib.

The Real˙ZF and Real˙ZF˙1 theories contain the construction of real numbers based on the paper [2] by R. D. Arthan (not Cauchy sequences, not Dedekind sections). The heavy lifting is done mostly in Group˙ZF˙3, Ring˙ZF˙1 and Int˙ZF˙2. Real˙ZF contains the part of the construction that can be done starting from generic abelian groups (rather than additive group of integers). This allows to show that real numbers form a ring. Real˙ZF˙1 continues the construction using properties specific to the integers and showing that real numbers constructed this way form a complete ordered field.

Cardinal˙ZF provides a couple of theorems about cardinals that are mostly used for studying properties of topological properties (yes, this is kind of meta). The main result (proven without AC) is that if two sets can be injectively mapped into an infinite cardinal, then so can be their union. There is also a definition of the Axiom of Choice specific for a given cardinal (so that the choice function exists for families of sets of given cardinality). Some properties are proven for such predicates, like that for finite families of sets the choice function always exists (in ZF) and that the axiom of choice for a larger cardinal implies one for a smaller cardinal.

Group˙ZF˙4 considers conjugate of subgroup and defines simple groups. A nice theorem here is that endomorphisms of an abelian group form a ring. The first isomorphism theorem (a group homomorphism h induces an isomorphism between the group divided by the kernel of h and the image of h) is proven.

Turns out given a property of a topological space one can define a local version of a property in general. This is studied in the the Topology˙ZF˙properties˙2 theory and applied to local versions of the property of being finite or compact or Hausdorff (i.e. locally finite, locally compact, locally Hausdorff). There are a couple of nice applications, like one-point compactification that allows to show that every locally compact Hausdorff space is regular. Also there are some results on the interplay between hereditability of a property and local properties.

For a given surjection $f : X \to Y$, where X is a topological space one can consider the weakest topology on Y which makes f continuous, let's call it a quotient topology generated by f. The quotient topology generated by an equivalence relation r on X is actually a special case of this setup, where f is the natural projection of X on the quotient X/r. The properties of these two ways of getting new topologies are studied in Topology˙ZF˙8 theory. The main result is that any quotient topology generated by a function is homeomorphic to a topology given by an equivalence relation, so these two approaches to quotient topologies are kind of equivalent.

As we all know, automorphisms of a topological space form a group. This fact is proven in Topology˙ZF˙9 and the automorphism groups for co-cardinal, included-set, and excluded-set topologies are identified. For order topologies it is shown that order isomorphisms are homeomorphisms of the topology induced by the order. Properties preserved by continuous functions are studied and as an application it is shown for example that quotient topological spaces of compact (or connected) spaces are compact (or connected, resp.)

The Topology˙ZF˙10 theory is about products of two topological spaces. It is proven that if two spaces are T_0 (or T_1, T_2, regular, connected) then their product is as well.

Given a total order on a set one can define a natural topology on it generated by taking the rays and intervals as the base. The Topology˙ZF˙11 theory studies relations between the order and various properties of generated topology. For example one can show that if the order topology is connected, then the order is complete (in the sense that for each set bounded from above the set of upper bounds has a minimum). For a given cardinal κ we can consider generalized notion of $\kappa - separability$. Turns out κ-separability is related to (order) density of sets of cardinality κ for order topologies.

Being a topological group imposes additional structure on the topology of the group, in particular its separation properties. In Topological˙Group˙ZF˙1.thy theory it is shown that if a topology is T_0, then it must be T_3 , and that the topology in a topological group is always regular.

For a given normal subgroup of a topological group we can define a topology on the quotient group in a natural way. At the end of the Topological˙Group˙ZF˙2.thy theory it is shown that such topology on the quotient group makes it a topological group.

The Topological˙Group˙ZF˙3.thy theory studies the topologies on subgroups of a topological group. A couple of nice basic properties are shown, like that the closure of a subgroup is a subgroup, closure of a normal subgroup is normal and, a bit more surprising (to me) property that every locally-compact subgroup of a T_0 group is closed.

In Complex_ZF we construct complex numbers starting from a complete ordered field (a model of real numbers). We also define the notation for writing about complex numbers and prove that the structure of complex numbers constructed there satisfies the axioms of complex numbers used in Metamath.

MMI_prelude defines the mmisar0 context in which most theorems translated from Metamath are proven. It also contains a chapter explaining how the translation works.

In the Metamath_interface theory we prove a theorem that the mmisar0 context is valid (can be used) in the complex0 context. All theories using the translated results will import the Metamath_interface theory. The Metamath_sampler theory provides some examples of using the translated theorems in the complex0 context.

The theories MMI_logic_and_sets, MMI_Complex, MMI_Complex_1 and MMI_Complex_2 contain the theorems imported from the Metamath's set.mm database. As the translated proofs are rather verbose these theories are not printed in this proof document. The full list of translated facts can be found in the Metamath_theorems.txt file included in the IsarMathLib distribution. The MMI_examples provides some theorems imported from Metamath that are printed in this proof document as examples of how translated proofs look like.

end

2 First Order Logic

theory Fol1 imports ZF.Trancl

begin

Isabelle/ZF builds on the first order logic. Almost everything one would like to have in this area is covered in the standard Isabelle libraries. The material in this theory provides some lemmas that are missing or allow for a more readable proof style.

2.1 Notions and lemmas in FOL

This section contains mostly shortcuts and workarounds that allow to use more readable coding style.

The next lemma serves as a workaround to problems with applying the definition of transitivity (of a relation) in our coding style (any attempt to do something like using trans˙def puts Isabelle in an infinite loop).

lemma Fol1˙L2: assumes
  A1: $\forall$ x y z. $\langle$x, y$\rangle \in$ r $\wedge$ $\langle$y, z$\rangle \in$ r $\longrightarrow$ $\langle$x, z$\rangle \in$ r
  shows trans(r)
$\langle$*proof*$\rangle$

Another workaround for the problem of Isabelle simplifier looping when the transitivity definition is used.

lemma Fol1˙L3: assumes A1: trans(r) and A2: $\langle$ a,b$\rangle \in$ r $\wedge$ $\langle$ b,c$\rangle \in$ r
  shows $\langle$ a,c$\rangle \in$ r
$\langle$*proof*$\rangle$

There is a problem with application of the definition of asymetry for relations. The next lemma is a workaround.

lemma Fol1˙L4:
  assumes A1: antisym(r) and A2: $\langle$ a,b$\rangle \in$ r $\langle$ b,a$\rangle \in$ r
  shows a=b
$\langle$*proof*$\rangle$

The definition below implements a common idiom that states that (perhaps under some assumptions) exactly one of given three statements is true.

definition
  Exactly˙1˙of˙3˙holds(p,q,r) $\equiv$
  (p$\vee$q$\vee$r) $\wedge$ (p $\longrightarrow$ $\neg$q $\wedge$ $\neg$r) $\wedge$ (q $\longrightarrow$ $\neg$p $\wedge$ $\neg$r) $\wedge$ (r $\longrightarrow$ $\neg$p $\wedge$ $\neg$q)

The next lemma allows to prove statements of the form Exactly˙1˙of˙3˙holds(p,q,r).

lemma Fol1˙L5:
  assumes p$\vee$q$\vee$r
  and p $\longrightarrow$ $\neg$q $\wedge$ $\neg$r
  and q $\longrightarrow$ $\neg$p $\wedge$ $\neg$r
  and r $\longrightarrow$ $\neg$p $\wedge$ $\neg$q
  shows Exactly˙1˙of˙3˙holds(p,q,r)
$\langle$*proof*$\rangle$

If exactly one of p, q, r holds and p is not true, then q or r.

lemma Fol1˙L6:
  assumes A1: $\neg$p and A2: Exactly˙1˙of˙3˙holds(p,q,r)
  shows q$\vee$r
$\langle$*proof*$\rangle$

If exactly one of p, q, r holds and q is true, then r can not be true.

lemma Fol1˙L7:
  assumes A1: q and A2: Exactly˙1˙of˙3˙holds(p,q,r)
  shows $\neg$r
$\langle$*proof*$\rangle$

The next lemma demonstrates an elegant form of the Exactly˙1˙of˙3˙holds(p,q,r) predicate.

lemma Fol1˙L8:
  shows Exactly˙1˙of˙3˙holds(p,q,r) $\longleftrightarrow$ (p$\longleftrightarrow$q$\longleftrightarrow$r) $\wedge$ $\neg$(p$\wedge$q$\wedge$r)
⟨*proof*⟩

A property of the Exactly˙1˙of˙3˙holds predicate.

lemma Fol1˙L8A: assumes A1: Exactly˙1˙of˙3˙holds(p,q,r)
  shows p $\longleftrightarrow$ $\neg$(q $\vee$ r)
⟨*proof*⟩

Exclusive or definition. There is one also defined in the standard Isabelle, denoted xor, but it relates to boolean values, which are sets. Here we define a logical functor.

definition
  Xor (infixl Xor 66) where
  p Xor q $\equiv$ (p$\vee$q) $\wedge$ $\neg$(p $\wedge$ q)

The "exclusive or" is the same as negation of equivalence.

lemma Fol1˙L9: shows p Xor q $\longleftrightarrow$ $\neg$(p$\longleftrightarrow$q)
  ⟨*proof*⟩

Equivalence relations are symmetric.

lemma equiv˙is˙sym: assumes A1: equiv(X,r) and A2: $\langle$x,y$\rangle$ $\in$ r
  shows $\langle$y,x$\rangle$ $\in$ r
⟨*proof*⟩

end

3 ZF set theory basics

theory ZF1 imports ZF.equalities

begin

The standard Isabelle distribution contains lots of facts about basic set theory. This theory file adds some more.

3.1 Lemmas in Zermelo-Fraenkel set theory

Here we put lemmas from the set theory that we could not find in the standard Isabelle distribution.

If one collection is contained in another, then we can say the same about their unions.

lemma collection˙contain: assumes A⊆B shows $\bigcup$A ⊆ $\bigcup$B
⟨*proof*⟩

If all sets of a nonempty collection are the same, then its union is the same.

lemma ZF1˙1˙L1: assumes C≠0 and ∀y∈C. b(y) = A
shows ($\bigcup$y∈C. b(y)) = A ⟨*proof*⟩

The union af all values of a constant meta-function belongs to the same set as the constant.

lemma ZF1˙1˙L2: assumes A1:C≠0 and A2: ∀x∈C. b(x) ∈ A
and A3: ∀x y. x∈C ∧ y∈C ⟶ b(x) = b(y)
shows ($\bigcup$x∈C. b(x))∈A
⟨*proof*⟩

If two meta-functions are the same on a cartesian product, then the subsets defined by them are the same. I am surprised Isabelle can not handle this automatically.

lemma ZF1˙1˙L4: assumes A1: ∀x∈X.∀y∈Y. a(x,y) = b(x,y)
shows –a(x,y). ⟨x,y⟩ ∈ X×Y″ = –b(x,y). ⟨x,y⟩ ∈ X×Y″
⟨*proof*⟩

If two meta-functions are the same on a cartesian product, then the subsets defined by them are the same. This is similar to ZF1˙1˙L4, except that the set definition varies over p∈X×Y rather than ⟨ x,y⟩∈X×Y.

lemma ZF1˙1˙L4A: assumes A1: ∀x∈X.∀y∈Y. a(⟨ x,y⟩) = b(x,y)
shows –a(p). p ∈ X×Y″ = –b(x,y). ⟨x,y⟩ ∈ X×Y″
⟨*proof*⟩

A lemma about inclusion in cartesian products. Included here to remember that we need the $U \times V \neq \emptyset$ assumption.

lemma prod˙subset: assumes U×V≠0 U×V ⊆ X×Y shows U⊆X and V⊆Y
⟨*proof*⟩

A technical lemma about sections in cartesian products.

lemma section˙proj: assumes A ⊆ X×Y and U×V ⊆ A and x ∈ U y ∈ V
shows U ⊆ –t∈X. ⟨t,y⟩ ∈ A″ and V ⊆ –t∈Y. ⟨x,t⟩ ∈ A″
⟨*proof*⟩

If two meta-functions are the same on a set, then they define the same set by separation.

lemma ZF1˙1˙L4B: assumes ∀x∈X. a(x) = b(x)
shows –a(x). x∈X″ = –b(x). x∈X″
⟨*proof*⟩

A set defined by a constant meta-function is a singleton.

lemma ZF1˙1˙L5: assumes X≠0 and ∀x∈X. b(x) = c shows –b(x). x∈X'' = –c'' ⟨*proof*⟩

Most of the time, auto does this job, but there are strange cases when the next lemma is needed.

lemma subset˙with˙property: assumes Y = –x∈X. b(x)''
shows Y ⊆ X
⟨*proof*⟩

We can choose an element from a nonempty set.

lemma nonempty˙has˙element: assumes X≠0 shows ∃x. x∈X
⟨*proof*⟩

In Isabelle/ZF the intersection of an empty family is empty. This is exactly lemma Inter˙0 from Isabelle's equalities theory. We repeat this lemma here as it is very difficult to find. This is one reason we need comments before every theorem: so that we can search for keywords.

lemma inter˙empty˙empty: shows ⋂0 = 0 ⟨*proof*⟩

If an intersection of a collection is not empty, then the collection is not empty. We are (ab)using the fact the the intersection of empty collection is defined to be empty.

lemma inter˙nempty˙nempty: assumes ⋂A ≠ 0 shows A≠0
⟨*proof*⟩

For two collections S, T of sets we define the product collection as the collections of cartesian products $A \times B$, where $A \in S, B \in T$.

definition
ProductCollection(T,S) ≡ ⋃U∈T.–U×V. V∈S''

The union of the product collection of collections S, T is the cartesian product of $\bigcup S$ and $\bigcup T$.

lemma ZF1˙1˙L6: shows ⋃ ProductCollection(S,T) = ⋃S × ⋃T
⟨*proof*⟩

An intersection of subsets is a subset.

lemma ZF1˙1˙L7: assumes A1: I≠0 and A2: ∀i∈I. P(i) ⊆ X
shows (⋂i∈I. P(i)) ⊆ X
⟨*proof*⟩

Isabelle/ZF has a "THE" construct that allows to define an element if there is only one such that is satisfies given predicate. In pure ZF we can express something similar using the indentity proven below.

lemma ZF1˙1˙L8: shows ⋃ –x'' = x ⟨*proof*⟩

Some properties of singletons.

lemma ZF1˙1˙L9: assumes A1: ∃! x. x∈A ∧ φ(x)

shows
$\exists$ a. –x∈A. φ(x)″ = –a″
$\bigcup$ –x∈A. φ(x)″ ∈ A
φ($\bigcup$ –x∈A. φ(x)″)
⟨*proof*⟩

A simple version of ZF1˙1˙L9.

corollary singleton˙extract: assumes $\exists$! x. x∈A
shows ($\bigcup$ A) ∈ A
⟨*proof*⟩

A criterion for when a set defined by comprehension is a singleton.

lemma singleton˙comprehension:
assumes A1: y∈X and A2: $\forall$x∈X. $\forall$y∈X. P(x) = P(y)
shows ($\bigcup$–P(x). x∈X″) = P(y)
⟨*proof*⟩

Adding an element of a set to that set does not change the set.

lemma set˙elem˙add: assumes x∈X shows X ∪ –x″ = X ⟨*proof*⟩

Here we define a restriction of a collection of sets to a given set. In romantic math this is typically denoted $X \cap M$ and means $\{X \cap A : A \in M\}$. Note there is also restrict(f, A) defined for relations in ZF.thy.

definition
RestrictedTo (infixl –restricted to″ 70) where
M –restricted to″ X ≡ –X ∩ A . A ∈ M″

A lemma on a union of a restriction of a collection to a set.

lemma union˙restrict:
shows $\bigcup$(M –restricted to″ X) = ($\bigcup$M) ∩ X
⟨*proof*⟩

Next we show a technical identity that is used to prove sufficiency of some condition for a collection of sets to be a base for a topology.

lemma ZF1˙1˙L10: assumes A1: $\forall$U∈C. $\exists$A∈B. U = $\bigcup$A
shows $\bigcup\bigcup$ –$\bigcup$–A∈B. U = $\bigcup$A″. U∈C″ = $\bigcup$C
⟨*proof*⟩

Standard Isabelle uses a notion of cons(A,a) that can be thought of as $A \cup \{a\}$.

lemma consdef: shows cons(a,A) = A ∪ –a″
⟨*proof*⟩

If a difference between a set and a singleton is empty, then the set is empty or it is equal to the singleton.

lemma singl˙diff˙empty: assumes A - –x″ = 0
shows A = 0 ∨ A = –x″
⟨*proof*⟩

If a difference between a set and a singleton is the set, then the only element of the singleton is not in the set.

lemma singl˙diff˙eq: assumes A1: A - –x˝ = A
  shows x ∉ A
⟨*proof*⟩

A basic property of sets defined by comprehension.

lemma comprehension: assumes a ∈ –x∈X. p(x)˝
  shows a∈X and p(a) ⟨*proof*⟩

The image of a set by a greater relation is greater.

lemma image˙rel˙mono: assumes r⊆s shows r(A) ⊆ s(A)
  ⟨*proof*⟩

A technical lemma about relations: if x is in its image by a relation U and that image is contained in some set C, then the image of the singleton $\{x\}$ by the relation $U \cup C \times C$ equals C.

lemma image˙greater˙rel:
  assumes x ∈ U–x˝ and U–x˝ ⊆ C
  shows (U ∪ C×C)–x˝ = C
  ⟨*proof*⟩

It's hard to believe but there are cases where we have to reference this rule.

lemma set˙mem˙eq: assumes x∈A A=B shows x∈B ⟨*proof*⟩

end

4 Natural numbers in IsarMathLib

theory Nat˙ZF˙IML imports ZF.Arith

begin

The ZF set theory constructs natural numbers from the empty set and the notion of a one-element set. Namely, zero of natural numbers is defined as the empty set. For each natural number n the next natural number is defined as $n \cup \{n\}$. With this definition for every non-zero natural number we get the identity $n = \{0, 1, 2, .., n-1\}$. It is good to remember that when we see an expression like $f : n \to X$. Also, with this definition the relation "less or equal than" becomes "⊆" and the relation "less than" becomes "∈".

4.1 Induction

The induction lemmas in the standard Isabelle's Nat.thy file like for example nat˙induct require the induction step to be a higher order statement (the one that uses the $\Longrightarrow$ sign). I found it difficult to apply from Isar, which is perhaps more of an indication of my Isar skills than anything else. Anyway, here we provide a first order version that is easier to reference in Isar declarative style proofs.

The next theorem is a version of induction on natural numbers that I was thought in school.

theorem ind˙on˙nat:
  assumes A1: n∈nat and A2: P(0) and A3: ∀k∈nat. P(k)⟶P(succ(k))
  shows P(n)
⟨*proof*⟩

A nonzero natural number has a predecessor.

lemma Nat˙ZF˙1˙L3: assumes A1: n ∈ nat and A2: n≠0
  shows ∃k∈nat. n = succ(k)
⟨*proof*⟩

What is succ, anyway?

lemma succ˙explained: shows succ(n) = n ∪ –n˝
  ⟨*proof*⟩

Empty set is an element of every natural number which is not zero.

lemma empty˙in˙every˙succ: assumes A1: n ∈ nat
  shows 0 ∈ succ(n)
⟨*proof*⟩

If one natural number is less than another then their successors are in the same relation.

lemma succ˙ineq: assumes A1: n ∈ nat
  shows ∀i ∈ n. succ(i) ∈ succ(n)
⟨*proof*⟩

For natural numbers if $k \subseteq n$ the similar holds for their successors.

lemma succ˙subset: assumes A1: k ∈ nat n ∈ nat and A2: k⊆n
  shows succ(k) ⊆ succ(n)
⟨*proof*⟩

For any two natural numbers one of them is contained in the other.

lemma nat˙incl˙total: assumes A1: i ∈ nat j ∈ nat
  shows i ⊆ j ∨ j ⊆ i
⟨*proof*⟩

The set of natural numbers is the union of all successors of natural numbers.

lemma nat˙union˙succ: shows nat = (⋃n ∈ nat. succ(n))
⟨*proof*⟩

Successors of natural numbers are subsets of the set of natural numbers.

lemma succnat˙subset˙nat: assumes A1: n $\in$ nat shows succ(n) $\subseteq$ nat
$\langle proof \rangle$

Element of a natural number is a natural number.

lemma elem˙nat˙is˙nat: assumes A1: n $\in$ nat and A2: k$\in$n
shows k ¡ n k $\in$ nat k $\leq$ n $\langle$k,n$\rangle$ $\in$ Le
$\langle proof \rangle$

The set of natural numbers is the union of its elements.

lemma nat˙union˙nat: shows nat = $\bigcup$ nat
$\langle proof \rangle$

A natural number is a subset of the set of natural numbers.

lemma nat˙subset˙nat: assumes A1: n $\in$ nat shows n $\subseteq$ nat
$\langle proof \rangle$

Adding natural numbers does not decrease what we add to.

lemma add˙nat˙le: assumes A1: n $\in$ nat and A2: k $\in$ nat
shows
n $\leq$ n #+ k
n $\subseteq$ n #+ k
n $\subseteq$ k #+ n
$\langle proof \rangle$

Result of adding an element of k is smaller than of adding k.

lemma add˙lt˙mono:
assumes k $\in$ nat and j$\in$k
shows
(n #+ j) ¡ (n #+ k)
(n #+ j) $\in$ (n #+ k)
$\langle proof \rangle$

A technical lemma about a decomposition of a sum of two natural numbers: if a number i is from $m + n$ then it is either from m or can be written as a sum of m and a number from n. The proof by induction w.r.t. to m seems to be a bit heavy-handed, but I could not figure out how to do this directly from results from standard Isabelle/ZF.

lemma nat˙sum˙decomp: assumes A1: n $\in$ nat and A2: m $\in$ nat
shows $\forall$i $\in$ m #+ n. i $\in$ m $\vee$ ($\exists$j $\in$ n. i = m #+ j)
$\langle proof \rangle$

A variant of induction useful for finite sequences.

lemma fin˙nat˙ind: assumes A1: n $\in$ nat and A2: k $\in$ succ(n)
and A3: P(0) and A4: $\forall$j$\in$n. P(j) $\longrightarrow$ P(succ(j))
shows P(k)
$\langle proof \rangle$

Some properties of positive natural numbers.

lemma succ˙plus: assumes n ∈ nat k ∈ nat
  shows
  succ(n #+ j) ∈ nat
  succ(n) #+ succ(j) = succ(succ(n #+ j))
  ⟨*proof*⟩

4.2 Intervals

In this section we consider intervals of natural numbers i.e. sets of the form $\{n + j : j \in 0..k - 1\}$.

The interval is determined by two parameters: starting point and length. Recall that in standard Isabelle's Arith.thy the symbol #+ is defined as the sum of natural numbers.

definition

  NatInterval(n,k) ≡ –n #+ j. j∈k″

Subtracting the beginning af the interval results in a number from the length of the interval. It may sound weird, but note that the length of such interval is a natural number, hence a set.

lemma inter˙diff˙in˙len:
  assumes A1: k ∈ nat and A2: i ∈ NatInterval(n,k)
  shows i #- n ∈ k
⟨*proof*⟩

Intervals don't overlap with their starting point and the union of an interval with its starting point is the sum of the starting point and the length of the interval.

lemma length˙start˙decomp: assumes A1: n ∈ nat k ∈ nat
  shows
  n ∩ NatInterval(n,k) = 0
  n ∪ NatInterval(n,k) = n #+ k
⟨*proof*⟩

Sme properties of three adjacent intervals.

lemma adjacent˙intervals3: assumes n ∈ nat k ∈ nat m ∈ nat
  shows
  n #+ k #+ m = (n #+ k) ∪ NatInterval(n #+ k,m)
  n #+ k #+ m = n ∪ NatInterval(n,k #+ m)
  n #+ k #+ m = n ∪ NatInterval(n,k) ∪ NatInterval(n #+ k,m)
  ⟨*proof*⟩

end

5 Order relations - introduction

theory Order˙ZF imports Fol1

begin

This theory file considers various notion related to order. We redefine the notions of a total order, linear order and partial order to have the same terminology as Wikipedia (I found it very consistent across different areas of math). We also define and study the notions of intervals and bounded sets. We show the inclusion relations between the intervals with endpoints being in certain order. We also show that union of bounded sets are bounded. This allows to show in Finite˙ZF.thy that finite sets are bounded.

5.1 Definitions

In this section we formulate the definitions related to order relations.

A relation r is "total" on a set X if for all elements a, b of X we have a is in relation with b or b is in relation with a. An example is the $\leq$ relation on numbers.

definition
 IsTotal (infixl –is total on″ 65) where
 r –is total on″ X ≡ (∀a∈X.∀b∈X. ⟨ a,b⟩ ∈ r ∨ ⟨ b,a⟩ ∈ r)

A relation r is a partial order on X if it is reflexive on X (i.e. $\langle x, x\rangle$ for every $x \in X$), antisymmetric (if $\langle x, y\rangle \in r$ and $\langle y, x\rangle \in r$, then $x = y$) and transitive $\langle x, y\rangle \in r$ and $\langle y, z\rangle \in r$ implies $\langle x, z\rangle \in r$).

definition
 IsPartOrder(X,r) ≡ (refl(X,r) ∧ antisym(r) ∧ trans(r))

We define a linear order as a binary relation that is antisymmetric, transitive and total. Note that this terminology is different than the one used the standard Order.thy file.

definition
 IsLinOrder(X,r) ≡ (antisym(r) ∧ trans(r) ∧ (r –is total on″ X))

A set is bounded above if there is that is an upper bound for it, i.e. there are some u such that $\langle x, u\rangle \in r$ for all $x \in A$. In addition, the empty set is defined as bounded.

definition
 IsBoundedAbove(A,r) ≡ (A=0 ∨ (∃u. ∀x∈A. ⟨ x,u⟩ ∈ r))

We define sets bounded below analogously.

definition
 IsBoundedBelow(A,r) ≡ (A=0 ∨ (∃l. ∀x∈A. ⟨ l,x⟩ ∈ r))

A set is bounded if it is bounded below and above.

definition

IsBounded(A,r) ≡ (IsBoundedAbove(A,r) ∧ IsBoundedBelow(A,r))

The notation for the definition of an interval may be mysterious for some readers, see lemma Order˙ZF˙2˙L1 for more intuitive notation.

definition
Interval(r,a,b) ≡ r–a″ ∩ r-–b″

We also define the maximum (the greater of) two elemnts in the obvious way.

definition
GreaterOf(r,a,b) ≡ (if ⟨ a,b⟩ ∈ r then b else a)

The definition a a minimum (the smaller of) two elements.

definition
SmallerOf(r,a,b) ≡ (if ⟨ a,b⟩ ∈ r then a else b)

We say that a set has a maximum if it has an element that is not smaller that any other one. We show that under some conditions this element of the set is unique (if exists).

definition
HasAmaximum(r,A) ≡ ∃M∈A.∀x∈A. ⟨x,M⟩ ∈ r

A similar definition what it means that a set has a minimum.

definition
HasAminimum(r,A) ≡ ∃m∈A.∀x∈A. ⟨m,x⟩ ∈ r

Definition of the maximum of a set.

definition
Maximum(r,A) ≡ THE M. M∈A ∧ (∀x∈A. ⟨x,M⟩ ∈ r)

Definition of a minimum of a set.

definition
Minimum(r,A) ≡ THE m. m∈A ∧ (∀x∈A. ⟨m,x⟩ ∈ r)

The supremum of a set A is defined as the minimum of the set of upper bounds, i.e. the set $\{u.\forall_{a\in A}\langle a,u\rangle \in r\} = \bigcap_{a\in A} r\{a\}$. Recall that in Isabelle/ZF r-(A) denotes the inverse image of the set A by relation r (i.e. r-(A)=$\{x : \langle x,y\rangle \in r$ for some $y \in A\}$).

definition
Supremum(r,A) ≡ Minimum(r,⋂a∈A. r–a″)

The notion of "having a supremum" is the same as the set of upper bounds having a minimum, but having it a a separate notion does simplify notattion in soma cases. The definition is written in terms of images of singletons $\{x\}$ under relation. To understand this formulation note that the set of upper bounds of a set $A \subseteq X$ is $\bigcap_{x\in A}\{y \in X | \langle x,y\rangle \in r\}$, which is the same as

$\bigcap_{x\in A} r(\{x\})$, where $r(\{x\})$ is the image of the singleton $\{x\}$ under relation r.

definition
HasAsupremum(r,A) ≡ HasAminimum(r,⋂ a∈A. r–a″)

The notion of "having an infimum" is the same as the set of lower bounds having a maximum.

definition
HasAnInfimum(r,A) ≡ HasAmaximum(r,⋂ a∈A. r-–a″)

Infimum is defined analogously.

definition
Infimum(r,A) ≡ Maximum(r,⋂ a∈A. r-–a″)

We define a relation to be complete if every nonempty bounded above set has a supremum.

definition
IsComplete (˙ –is complete″) where
r –is complete″ ≡
∀ A. IsBoundedAbove(A,r) ∧ A≠0 ⟶ HasAminimum(r,⋂ a∈A. r–a″)

The essential condition to show that a total relation is reflexive.

lemma Order˙ZF˙1˙L1: assumes r –is total on″ X and a∈X
shows ⟨a,a⟩ ∈ r ⟨*proof*⟩

A total relation is reflexive.

lemma total˙is˙refl:
assumes r –is total on″ X
shows refl(X,r) ⟨*proof*⟩

A linear order is partial order.

lemma Order˙ZF˙1˙L2: assumes IsLinOrder(X,r)
shows IsPartOrder(X,r)
⟨*proof*⟩

Partial order that is total is linear.

lemma Order˙ZF˙1˙L3:
assumes IsPartOrder(X,r) and r –is total on″ X
shows IsLinOrder(X,r)
⟨*proof*⟩

Relation that is total on a set is total on any subset.

lemma Order˙ZF˙1˙L4: assumes r –is total on″ X and A⊆X
shows r –is total on″ A
⟨*proof*⟩

A linear relation is linear on any subset.

lemma ord˙linear˙subset: assumes IsLinOrder(X,r) and A⊆X
  shows IsLinOrder(A,r)
  ⟨*proof*⟩

If the relation is total, then every set is a union of those elements that are nongreater than a given one and nonsmaller than a given one.

lemma Order˙ZF˙1˙L5:
  assumes r –is total on″ X and A⊆X and a∈X
  shows A = –x∈A. ⟨x,a⟩ ∈ r″ ∪ –x∈A. ⟨a,x⟩ ∈ r″
  ⟨*proof*⟩

A technical fact about reflexive relations.

lemma refl˙add˙point:
  assumes refl(X,r) and A ⊆ B ∪ –x″ and B ⊆ X and
  x ∈ X and ∀y∈B. ⟨y,x⟩ ∈ r
  shows ∀a∈A. ⟨a,x⟩ ∈ r
  ⟨*proof*⟩

5.2 Intervals

In this section we discuss intervals.

The next lemma explains the notation of the definition of an interval.

lemma Order˙ZF˙2˙L1:
  shows x ∈ Interval(r,a,b) ⟷ ⟨ a,x⟩ ∈ r ∧ ⟨ x,b⟩ ∈ r
  ⟨*proof*⟩

Since there are some problems with applying the above lemma (seems that simp and auto don't handle equivalence very well), we split Order˙ZF˙2˙L1 into two lemmas.

lemma Order˙ZF˙2˙L1A: assumes x ∈ Interval(r,a,b)
  shows ⟨ a,x⟩ ∈ r ⟨ x,b⟩ ∈ r
  ⟨*proof*⟩

Order˙ZF˙2˙L1, implication from right to left.

lemma Order˙ZF˙2˙L1B: assumes ⟨ a,x⟩ ∈ r ⟨ x,b⟩ ∈ r
  shows x ∈ Interval(r,a,b)
  ⟨*proof*⟩

If the relation is reflexive, the endpoints belong to the interval.

lemma Order˙ZF˙2˙L2: assumes refl(X,r)
  and a∈X b∈X and ⟨ a,b⟩ ∈ r
  shows
  a ∈ Interval(r,a,b)
  b ∈ Interval(r,a,b)
  ⟨*proof*⟩

Under the assumptions of Order˙ZF˙2˙L2, the interval is nonempty.

lemma Order˙ZF˙2˙L2A: assumes refl(X,r)
and a∈X b∈X and ⟨ a,b⟩ ∈ r
shows Interval(r,a,b) ≠ 0
⟨*proof*⟩

If a, b, c, d are in this order, then $[b, c] \subseteq [a, d]$. We only need trasitivity for this to be true.

lemma Order˙ZF˙2˙L3:
assumes A1: trans(r) and A2:⟨ a,b⟩∈r ⟨ b,c⟩∈r ⟨ c,d⟩∈r
shows Interval(r,b,c) ⊆ Interval(r,a,d)
⟨*proof*⟩

For reflexive and antisymmetric relations the interval with equal endpoints consists only of that endpoint.

lemma Order˙ZF˙2˙L4:
assumes A1: refl(X,r) and A2: antisym(r) and A3: a∈X
shows Interval(r,a,a) = –a″
⟨*proof*⟩

For transitive relations the endpoints have to be in the relation for the interval to be nonempty.

lemma Order˙ZF˙2˙L5: assumes A1: trans(r) and A2: ⟨ a,b⟩ ∉ r
shows Interval(r,a,b) = 0
⟨*proof*⟩

If a relation is defined on a set, then intervals are subsets of that set.

lemma Order˙ZF˙2˙L6: assumes A1: r ⊆ X×X
shows Interval(r,a,b) ⊆ X
⟨*proof*⟩

5.3 Bounded sets

In this section we consider properties of bounded sets.

For reflexive relations singletons are bounded.

lemma Order˙ZF˙3˙L1: assumes refl(X,r) and a∈X
shows IsBounded(–a″,r)
⟨*proof*⟩

Sets that are bounded above are contained in the domain of the relation.

lemma Order˙ZF˙3˙L1A: assumes r ⊆ X×X
and IsBoundedAbove(A,r)
shows A⊆X ⟨*proof*⟩

Sets that are bounded below are contained in the domain of the relation.

lemma Order˙ZF˙3˙L1B: assumes r ⊆ X×X
and IsBoundedBelow(A,r)

shows A⊆X ⟨*proof*⟩

For a total relation, the greater of two elements, as defined above, is indeed greater of any of the two.

lemma Order˙ZF˙3˙L2: assumes r –is total on″ X
and x∈X y∈X
shows
⟨x,GreaterOf(r,x,y)⟩ ∈ r
⟨y,GreaterOf(r,x,y)⟩ ∈ r
⟨SmallerOf(r,x,y),x⟩ ∈ r
⟨SmallerOf(r,x,y),y⟩ ∈ r
⟨*proof*⟩

If A is bounded above by u, B is bounded above by w, then $A \cup B$ is bounded above by the greater of u, w.

lemma Order˙ZF˙3˙L2B:
assumes A1: r –is total on″ X and A2: trans(r)
and A3: u∈X w∈X
and A4: ∀x∈A. ⟨ x,u⟩ ∈ r ∀x∈B. ⟨ x,w⟩ ∈ r
shows ∀x∈A∪B. ⟨x,GreaterOf(r,u,w)⟩ ∈ r
⟨*proof*⟩

For total and transitive relation the union of two sets bounded above is bounded above.

lemma Order˙ZF˙3˙L3:
assumes A1: r –is total on″ X and A2: trans(r)
and A3: IsBoundedAbove(A,r) IsBoundedAbove(B,r)
and A4: r ⊆ X×X
shows IsBoundedAbove(A∪B,r)
⟨*proof*⟩

For total and transitive relations if a set A is bounded above then $A \cup \{a\}$ is bounded above.

lemma Order˙ZF˙3˙L4:
assumes A1: r –is total on″ X and A2: trans(r)
and A3: IsBoundedAbove(A,r) and A4: a∈X and A5: r ⊆ X×X
shows IsBoundedAbove(A∪–a″,r)
⟨*proof*⟩

If A is bounded below by l, B is bounded below by m, then $A \cup B$ is bounded below by the smaller of u, w.

lemma Order˙ZF˙3˙L5B:
assumes A1: r –is total on″ X and A2: trans(r)
and A3: l∈X m∈X
and A4: ∀x∈A. ⟨ l,x⟩ ∈ r ∀x∈B. ⟨ m,x⟩ ∈ r
shows ∀x∈A∪B. ⟨SmallerOf(r,l,m),x⟩ ∈ r
⟨*proof*⟩

For total and transitive relation the union of two sets bounded below is bounded below.

lemma Order˙ZF˙3˙L6:
  assumes A1: r –is total on˝ X and A2: trans(r)
  and A3: IsBoundedBelow(A,r) IsBoundedBelow(B,r)
  and A4: r ⊆ X×X
  shows IsBoundedBelow(A∪B,r)
⟨*proof*⟩

For total and transitive relations if a set A is bounded below then $A \cup \{a\}$ is bounded below.

lemma Order˙ZF˙3˙L7:
  assumes A1: r –is total on˝ X and A2: trans(r)
  and A3: IsBoundedBelow(A,r) and A4: a∈X and A5: r ⊆ X×X
  shows IsBoundedBelow(A∪–a˝,r)
⟨*proof*⟩

For total and transitive relations unions of two bounded sets are bounded.

theorem Order˙ZF˙3˙T1:
  assumes r –is total on˝ X and trans(r)
  and IsBounded(A,r) IsBounded(B,r)
  and r ⊆ X×X
  shows IsBounded(A∪B,r)
  ⟨*proof*⟩

For total and transitive relations if a set A is bounded then $A \cup \{a\}$ is bounded.

lemma Order˙ZF˙3˙L8:
  assumes r –is total on˝ X and trans(r)
  and IsBounded(A,r) and a∈X and r ⊆ X×X
  shows IsBounded(A∪–a˝,r)
  ⟨*proof*⟩

A sufficient condition for a set to be bounded below.

lemma Order˙ZF˙3˙L9: assumes A1: ∀a∈A. ⟨l,a⟩ ∈ r
  shows IsBoundedBelow(A,r)
⟨*proof*⟩

A sufficient condition for a set to be bounded above.

lemma Order˙ZF˙3˙L10: assumes A1: ∀a∈A. ⟨a,u⟩ ∈ r
  shows IsBoundedAbove(A,r)
⟨*proof*⟩

Intervals are bounded.

lemma Order˙ZF˙3˙L11: shows
  IsBoundedAbove(Interval(r,a,b),r)
  IsBoundedBelow(Interval(r,a,b),r)

IsBounded(Interval(r,a,b),r)
⟨*proof*⟩

A subset of a set that is bounded below is bounded below.

lemma Order˙ZF˙3˙L12: assumes A1: IsBoundedBelow(A,r) and A2: B⊆A
shows IsBoundedBelow(B,r)
⟨*proof*⟩

A subset of a set that is bounded above is bounded above.

lemma Order˙ZF˙3˙L13: assumes A1: IsBoundedAbove(A,r) and A2: B⊆A
shows IsBoundedAbove(B,r)
⟨*proof*⟩

If for every element of X we can find one in A that is greater, then the A can not be bounded above. Works for relations that are total, transitive and antisymmetric, (i.e. for linear order relations).

lemma Order˙ZF˙3˙L14:
assumes A1: r –is total on˝ X
and A2: trans(r) and A3: antisym(r)
and A4: r ⊆ X×X and A5: X≠0
and A6: ∀x∈X. ∃a∈A. x≠a ∧ ⟨x,a⟩ ∈ r
shows ¬IsBoundedAbove(A,r)
⟨*proof*⟩

The set of elements in a set A that are nongreater than a given element is bounded above.

lemma Order˙ZF˙3˙L15: shows IsBoundedAbove(–x∈A. ⟨x,a⟩ ∈ r˝,r)
⟨*proof*⟩

If A is bounded below, then the set of elements in a set A that are nongreater than a given element is bounded.

lemma Order˙ZF˙3˙L16: assumes A1: IsBoundedBelow(A,r)
shows IsBounded(–x∈A. ⟨x,a⟩ ∈ r˝,r)
⟨*proof*⟩

end

6 More on order relations

theory Order˙ZF˙1 imports ZF.Order ZF1

begin

In Order˙ZF we define some notions related to order relations based on the nonstrict orders (≤ type). Some people however prefer to talk about these notions in terms of the strict order relation (< type). This is the case for the standard Isabelle Order.thy and also for Metamath. In this theory file we repeat some developments from Order˙ZF using the strict order relation as a basis.This is mostly useful for Metamath translation, but is also of some general interest. The names of theorems are copied from Metamath.

6.1 Definitions and basic properties

In this section we introduce some definitions taken from Metamath and relate them to the ones used by the standard Isabelle Order.thy.

The next definition is the strict version of the linear order. What we write as R Orders A is written *ROrdA* in Metamath.

definition
StrictOrder (infix Orders 65) where
  R Orders A ≡ ∀x y z. (x∈A ∧ y∈A ∧ z∈A) ⟶
  (⟨x,y⟩ ∈ R ⟷ ¬(x=y ∨ ⟨y,x⟩ ∈ R)) ∧
  (⟨x,y⟩ ∈ R ∧ ⟨y,z⟩ ∈ R ⟶ ⟨x,z⟩ ∈ R)

The definition of supremum for a (strict) linear order.

definition
  Sup(B,A,R) ≡
  ⋃ –x ∈ A. (∀y∈B. ⟨x,y⟩ ∉ R) ∧
  (∀y∈A. ⟨y,x⟩ ∈ R ⟶ (∃z∈B. ⟨y,z⟩ ∈ R))″

Definition of infimum for a linear order. It is defined in terms of supremum.

definition
  Infim(B,A,R) ≡ Sup(B,A,converse(R))

If relation R orders a set A, (in Metamath sense) then R is irreflexive, transitive and linear therefore is a total order on A (in Isabelle sense).

lemma orders˙imp˙tot˙ord: assumes A1: R Orders A
  shows
  irrefl(A,R)
  trans[A](R)
  part˙ord(A,R)
  linear(A,R)
  tot˙ord(A,R)
⟨*proof*⟩

A converse of orders˙imp˙tot˙ord. Together with that theorem this shows that Metamath's notion of an order relation is equivalent to Isabelles tot˙ord predicate.

lemma tot˙ord˙imp˙orders: assumes A1: tot˙ord(A,R)
  shows R Orders A
⟨*proof*⟩

6.2 Properties of (strict) total orders

In this section we discuss the properties of strict order relations. This continues the development contained in the standard Isabelle's Order.thy with a view towards using the theorems translated from Metamath.

A relation orders a set iff the converse relation orders a set. Going one way we can use the the lemma tot˙od˙converse from the standard Isabelle's Order.thy.The other way is a bit more complicated (note that in Isabelle for converse(converse(r)) = r one needs r to consist of ordered pairs, which does not follow from the StrictOrder definition above).

lemma cnvso: shows R Orders A ⟷ converse(R) Orders A
⟨*proof*⟩

Supremum is unique, if it exists.

lemma supeu: assumes A1: R Orders A and A2: x∈A and
A3: ∀y∈B. ⟨x,y⟩ ∉ R and A4: ∀y∈A. ⟨y,x⟩ ∈ R ⟶ (∃z∈B. ⟨y,z⟩ ∈ R)
shows
∃!x. x∈A∧(∀y∈B. ⟨x,y⟩ ∉ R) ∧ (∀y∈A. ⟨y,x⟩ ∈ R ⟶ (∃z∈B. ⟨y,z⟩ ∈ R))
⟨*proof*⟩

Supremum has expected properties if it exists.

lemma sup˙props: assumes A1: R Orders A and
A2: ∃x∈A. (∀y∈B. ⟨x,y⟩ ∉ R) ∧ (∀y∈A. ⟨y,x⟩ ∈ R ⟶ (∃z∈B. ⟨y,z⟩ ∈ R))
shows
Sup(B,A,R) ∈ A
∀y∈B. ⟨Sup(B,A,R),y⟩ ∉ R
∀y∈A. ⟨y,Sup(B,A,R)⟩ ∈ R ⟶ (∃z∈B. ⟨y,z⟩ ∈ R)
⟨*proof*⟩

Elements greater or equal than any element of B are greater or equal than supremum of B.

lemma supnub: assumes A1: R Orders A and A2:
∃x∈A. (∀y∈B. ⟨x,y⟩ ∉ R) ∧ (∀y∈A. ⟨y,x⟩ ∈ R ⟶ (∃z∈B. ⟨y,z⟩ ∈ R))
and A3: c ∈ A and A4: ∀z∈B. ⟨c,z⟩ ∉ R
shows ⟨c, Sup(B,A,R)⟩ ∉ R
⟨*proof*⟩

end

7 Even more on order relations

theory Order˙ZF˙1a imports Order˙ZF

begin

This theory is a continuation of Order_ZF and talks about maximuma and minimum of a set, supremum and infimum and strict (not reflexive) versions of order relations.

7.1 Maximum and minimum of a set

In this section we show that maximum and minimum are unique if they exist. We also show that union of sets that have maxima (minima) has a maximum (minimum). We also show that singletons have maximum and minimum. All this allows to show (in Finite_ZF) that every finite set has well-defined maximum and minimum.

A somewhat technical fact that allows to reduce the number of premises in some theorems: the assumption that a set has a maximum implies that it is not empty.

lemma set_max_not_empty: assumes HasAmaximum(r,A) shows A≠0
⟨*proof*⟩

If a set has a maximum implies that it is not empty.

lemma set_min_not_empty: assumes HasAminimum(r,A) shows A≠0
⟨*proof*⟩

If a set has a supremum then it cannot be empty. We are probably using the fact that $\bigcap \emptyset = \emptyset$, which makes me a bit anxious as this I think is just a convention.

lemma set_sup_not_empty: assumes HasAsupremum(r,A) shows A≠0
⟨*proof*⟩

If a set has an infimum then it cannot be empty.

lemma set_inf_not_empty: assumes HasAnInfimum(r,A) shows A≠0
⟨*proof*⟩

For antisymmetric relations maximum of a set is unique if it exists.

lemma Order_ZF_4_L1: assumes A1: antisym(r) and A2: HasAmaximum(r,A)
shows ∃!M. M∈A ∧ (∀x∈A. ⟨ x,M⟩ ∈ r)
⟨*proof*⟩

For antisymmetric relations minimum of a set is unique if it exists.

lemma Order_ZF_4_L2: assumes A1: antisym(r) and A2: HasAminimum(r,A)
shows ∃!m. m∈A ∧ (∀x∈A. ⟨ m,x⟩ ∈ r)
⟨*proof*⟩

Maximum of a set has desired properties.

lemma Order_ZF_4_L3: assumes A1: antisym(r) and A2: HasAmaximum(r,A)
shows Maximum(r,A) ∈ A ∀x∈A. ⟨x,Maximum(r,A)⟩ ∈ r

⟨*proof*⟩

Minimum of a set has desired properties.

lemma Order˙ZF˙4˙L4: assumes A1: antisym(r) and A2: HasAminimum(r,A)
 shows Minimum(r,A) $\in$ A $\forall$x$\in$A. ⟨Minimum(r,A),x⟩ $\in$ r
⟨*proof*⟩

For total and transitive relations a union a of two sets that have maxima has a maximum.

lemma Order˙ZF˙4˙L5:
 assumes A1: r –is total on" (A$\cup$B) and A2: trans(r)
 and A3: HasAmaximum(r,A) HasAmaximum(r,B)
 shows HasAmaximum(r,A$\cup$B)
⟨*proof*⟩

For total and transitive relations A union a of two sets that have minima has a minimum.

lemma Order˙ZF˙4˙L6:
 assumes A1: r –is total on" (A$\cup$B) and A2: trans(r)
 and A3: HasAminimum(r,A) HasAminimum(r,B)
 shows HasAminimum(r,A$\cup$B)
⟨*proof*⟩

Set that has a maximum is bounded above.

lemma Order˙ZF˙4˙L7:
 assumes HasAmaximum(r,A)
 shows IsBoundedAbove(A,r)
 ⟨*proof*⟩

Set that has a minimum is bounded below.

lemma Order˙ZF˙4˙L8A:
 assumes HasAminimum(r,A)
 shows IsBoundedBelow(A,r)
 ⟨*proof*⟩

For reflexive relations singletons have a minimum and maximum.

lemma Order˙ZF˙4˙L8: assumes refl(X,r) and a$\in$X
 shows HasAmaximum(r,–a") HasAminimum(r,–a")
 ⟨*proof*⟩

For total and transitive relations if we add an element to a set that has a maximum, the set still has a maximum.

lemma Order˙ZF˙4˙L9:
 assumes A1: r –is total on" X and A2: trans(r)
 and A3: A$\subseteq$X and A4: a$\in$X and A5: HasAmaximum(r,A)
 shows HasAmaximum(r,A$\cup$–a")
⟨*proof*⟩

For total and transitive relations if we add an element to a set that has a minimum, the set still has a minimum.

lemma Order˙ZF˙4˙L10:
  assumes A1: r –is total on˝ X and A2: trans(r)
  and A3: A⊆X and A4: a∈X and A5: HasAminimum(r,A)
  shows HasAminimum(r,A∪–a˝)
⟨*proof*⟩

If the order relation has a property that every nonempty bounded set attains a minimum (for example integers are like that), then every nonempty set bounded below attains a minimum.

lemma Order˙ZF˙4˙L11:
  assumes A1: r –is total on˝ X and
  A2: trans(r) and
  A3: r ⊆ X×X and
  A4: ∀A. IsBounded(A,r) ∧ A≠0 ⟶ HasAminimum(r,A) and
  A5: B≠0 and A6: IsBoundedBelow(B,r)
  shows HasAminimum(r,B)
⟨*proof*⟩

A dual to Order˙ZF˙4˙L11: If the order relation has a property that every nonempty bounded set attains a maximum (for example integers are like that), then every nonempty set bounded above attains a maximum.

lemma Order˙ZF˙4˙L11A:
  assumes A1: r –is total on˝ X and
  A2: trans(r) and
  A3: r ⊆ X×X and
  A4: ∀A. IsBounded(A,r) ∧ A≠0 ⟶ HasAmaximum(r,A) and
  A5: B≠0 and A6: IsBoundedAbove(B,r)
  shows HasAmaximum(r,B)
⟨*proof*⟩

If a set has a minimum and L is less or equal than all elements of the set, then L is less or equal than the minimum.

lemma Order˙ZF˙4˙L12:
  assumes antisym(r) and HasAminimum(r,A) and ∀a∈A. ⟨L,a⟩ ∈ r
  shows ⟨L,Minimum(r,A)⟩ ∈ r
  ⟨*proof*⟩

If a set has a maximum and all its elements are less or equal than M, then the maximum of the set is less or equal than M.

lemma Order˙ZF˙4˙L13:
  assumes antisym(r) and HasAmaximum(r,A) and ∀a∈A. ⟨a,M⟩ ∈ r
  shows ⟨Maximum(r,A),M⟩ ∈ r
  ⟨*proof*⟩

If an element belongs to a set and is greater or equal than all elements of that set, then it is the maximum of that set.

lemma Order˙ZF˙4˙L14:
  assumes A1: antisym(r) and A2: M ∈ A and
  A3: ∀ a∈A. ⟨a,M⟩ ∈ r
  shows Maximum(r,A) = M
⟨*proof*⟩

If an element belongs to a set and is less or equal than all elements of that set, then it is the minimum of that set.

lemma Order˙ZF˙4˙L15:
  assumes A1: antisym(r) and A2: m ∈ A and
  A3: ∀ a∈A. ⟨m,a⟩ ∈ r
  shows Minimum(r,A) = m
⟨*proof*⟩

If a set does not have a maximum, then for any its element we can find one that is (strictly) greater.

lemma Order˙ZF˙4˙L16:
  assumes A1: antisym(r) and A2: r –is total on" X and
  A3: A⊆X and
  A4: ¬HasAmaximum(r,A) and
  A5: x∈A
  shows ∃ y∈A. ⟨x,y⟩ ∈ r ∧ y≠x
⟨*proof*⟩

7.2 Supremum and Infimum

In this section we consider the notions of supremum and infimum a set.

Elements of the set of upper bounds are indeed upper bounds. Isabelle also thinks it is obvious.

lemma Order˙ZF˙5˙L1: assumes u ∈ (⋂a∈A. r–a") and a∈A
  shows ⟨a,u⟩ ∈ r
  ⟨*proof*⟩

Elements of the set of lower bounds are indeed lower bounds. Isabelle also thinks it is obvious.

lemma Order˙ZF˙5˙L2: assumes l ∈ (⋂a∈A. r-–a") and a∈A
  shows ⟨l,a⟩ ∈ r
  ⟨*proof*⟩

If the set of upper bounds has a minimum, then the supremum is less or equal than any upper bound. We can probably do away with the assumption that A is not empty, (ab)using the fact that intersection over an empty family is defined in Isabelle to be empty. This lemma is obsolete and will be removed in the future. Use sup˙leq˙up˙bnd instead.

lemma Order˙ZF˙5˙L3: assumes A1: antisym(r) and A2: A≠0 and
  A3: HasAminimum(r,⋂a∈A. r–a") and

A4: ∀a∈A. ⟨a,u⟩ ∈ r
shows ⟨Supremum(r,A),u⟩ ∈ r
⟨*proof*⟩

Supremum is less or equal than any upper bound.

lemma sup˙leq˙up˙bnd: assumes antisym(r) HasAsupremum(r,A) ∀a∈A. ⟨a,u⟩ ∈ r
shows ⟨Supremum(r,A),u⟩ ∈ r
⟨*proof*⟩

Infimum is greater or equal than any lower bound. This lemma is obsolete and will be removed. Use inf˙geq˙lo˙bnd instead.

lemma Order˙ZF˙5˙L4: assumes A1: antisym(r) and A2: A≠0 and
A3: HasAmaximum(r,⋂a∈A. r-–a″) and
A4: ∀a∈A. ⟨l,a⟩ ∈ r
shows ⟨l,Infimum(r,A)⟩ ∈ r
⟨*proof*⟩

Infimum is greater or equal than any upper bound.

lemma inf˙geq˙lo˙bnd: assumes antisym(r) HasAnInfimum(r,A) ∀a∈A. ⟨u,a⟩ ∈ r
shows ⟨u,Infimum(r,A)⟩ ∈ r
⟨*proof*⟩

If z is an upper bound for A and is less or equal than any other upper bound, then z is the supremum of A.

lemma Order˙ZF˙5˙L5: assumes A1: antisym(r) and A2: A≠0 and
A3: ∀x∈A. ⟨x,z⟩ ∈ r and
A4: ∀y. (∀x∈A. ⟨x,y⟩ ∈ r) ⟶ ⟨z,y⟩ ∈ r
shows
HasAminimum(r,⋂a∈A. r–a″)
z = Supremum(r,A)
⟨*proof*⟩

The dual theorem to Order˙ZF˙5˙L5: if z is an lower bound for A and is greater or equal than any other lower bound, then z is the infimum of A.

lemma inf˙glb:
assumes antisym(r) A≠0 ∀x∈A. ⟨z,x⟩ ∈ r ∀y. (∀x∈A. ⟨y,x⟩ ∈ r) ⟶ ⟨y,z⟩ ∈ r
shows
HasAmaximum(r,⋂a∈A. r-–a″)
z = Infimum(r,A)
⟨*proof*⟩

Supremum and infimum of a singleton is the element.

lemma sup˙inf˙singl: assumes antisym(r) refl(X,r) z∈X
shows
HasAsupremum(r,–z″) Supremum(r,–z″) = z and
HasAnInfimum(r,–z″) Infimum(r,–z″) = z
⟨*proof*⟩

If a set has a maximum, then the maximum is the supremum. This lemma is obsolete, use max˙is˙sup instead.

lemma Order˙ZF˙5˙L6:
  assumes A1: antisym(r) and A2: A≠0 and
  A3: HasAmaximum(r,A)
  shows
  HasAminimum(r,⋂a∈A. r–a″)
  Maximum(r,A) = Supremum(r,A)
⟨*proof*⟩

Another version of Order˙ZF˙5˙L6 that: if a sat has a maximum then it has a supremum and the maximum is the supremum.

lemma max˙is˙sup: assumes antisym(r) A≠0 HasAmaximum(r,A)
  shows HasAsupremum(r,A) and Maximum(r,A) = Supremum(r,A)
⟨*proof*⟩

Minimum is the infimum if it exists.

lemma min˙is˙inf: assumes antisym(r) A≠0 HasAminimum(r,A)
  shows HasAnInfimum(r,A) and Minimum(r,A) = Infimum(r,A)
⟨*proof*⟩

For reflexive and total relations two-element set has a minimum and a maximum.

lemma min˙max˙two˙el: assumes refl(X,r) r –is total on″ X x∈X y∈X
  shows HasAminimum(r,–x,y″) and HasAmaximum(r,–x,y″)
  ⟨*proof*⟩

For antisymmetric, reflexive and total relations two-element set has a supremum and infimum.

lemma inf˙sup˙two˙el:assumes antisym(r) refl(X,r) r –is total on″ X x∈X y∈X
  shows
    HasAnInfimum(r,–x,y″)
    Minimum(r,–x,y″) = Infimum(r,–x,y″)
    HasAsupremum(r,–x,y″)
    Maximum(r,–x,y″) = Supremum(r,–x,y″)
  ⟨*proof*⟩

A sufficient condition for the supremum to be in the space.

lemma sup˙in˙space:
  assumes r ⊆ X×X antisym(r) HasAminimum(r,⋂a∈A. r–a″)
  shows Supremum(r,A) ∈ X and ∀x∈A. ⟨x,Supremum(r,A)⟩ ∈ r
⟨*proof*⟩

A sufficient condition for the infimum to be in the space.

lemma inf˙in˙space:
  assumes r ⊆ X×X antisym(r) HasAmaximum(r,⋂a∈A. r-–a″)
  shows Infimum(r,A) ∈ X and ∀x∈A. ⟨Infimum(r,A),x⟩ ∈ r

⟨*proof*⟩

Properties of supremum of a set for complete relations.

lemma Order˙ZF˙5˙L7:
  assumes A1: r ⊆ X×X and A2: antisym(r) and
  A3: r –is complete˝ and
  A4: A≠0 and A5: ∃x∈X. ∀y∈A. ⟨y,x⟩ ∈ r
  shows Supremum(r,A) ∈ X and ∀x∈A. ⟨x,Supremum(r,A)⟩ ∈ r
⟨*proof*⟩

Infimum of the set of infima of a collection of sets is infimum of the union.

lemma inf˙inf:
  assumes
    r ⊆ X×X antisym(r) trans(r)
    ∀T∈𝒯. HasAnInfimum(r,T)
    HasAnInfimum(r,–Infimum(r,T).T∈𝒯˝)
  shows
    HasAnInfimum(r,⋃𝒯) and Infimum(r,–Infimum(r,T).T∈𝒯˝) = Infimum(r,⋃𝒯)
⟨*proof*⟩

Supremum of the set of suprema of a collection of sets is supremum of the union.

lemma sup˙sup:
  assumes
    r ⊆ X×X antisym(r) trans(r)
    ∀T∈𝒯. HasAsupremum(r,T)
    HasAsupremum(r,–Supremum(r,T).T∈𝒯˝)
  shows
    HasAsupremum(r,⋃𝒯) and Supremum(r,–Supremum(r,T).T∈𝒯˝) = Supremum(r,⋃𝒯)
⟨*proof*⟩

If the relation is a linear order then for any element y smaller than the supremum of a set we can find one element of the set that is greater than y.

lemma Order˙ZF˙5˙L8:
  assumes A1: r ⊆ X×X and A2: IsLinOrder(X,r) and
  A3: r –is complete˝ and
  A4: A⊆X A≠0 and A5: ∃x∈X. ∀y∈A. ⟨y,x⟩ ∈ r and
  A6: ⟨y,Supremum(r,A)⟩ ∈ r   y ≠ Supremum(r,A)
  shows ∃z∈A. ⟨y,z⟩ ∈ r ∧ y ≠ z
⟨*proof*⟩

7.3 Strict versions of order relations

One of the problems with translating formalized mathematics from Metamath to IsarMathLib is that Metamath uses strict orders (of the < type) while in IsarMathLib we mostly use nonstrict orders (of the ≤ type). This doesn't really make any difference, but is annoying as we have to prove many theorems twice. In this section we prove some theorems to make it easier to translate the statements about strict orders to statements about the corresponding non-strict order and vice versa.

We define a strict version of a relation by removing the $y = x$ line from the relation.

definition
 StrictVersion(r) ≡ r - –⟨x,x⟩. x ∈ domain(r)″

A reformulation of the definition of a strict version of an order.

lemma def˙of˙strict˙ver: shows
 ⟨x,y⟩ ∈ StrictVersion(r) ⟷ ⟨x,y⟩ ∈ r ∧ x≠y
 ⟨*proof*⟩

The next lemma is about the strict version of an antisymmetric relation.

lemma strict˙of˙antisym:
 assumes A1: antisym(r) and A2: ⟨a,b⟩ ∈ StrictVersion(r)
 shows ⟨b,a⟩ ∉ StrictVersion(r)
⟨*proof*⟩

The strict version of totality.

lemma strict˙of˙tot:
 assumes r –is total on″ X and a∈X b∈X a≠b
 shows ⟨a,b⟩ ∈ StrictVersion(r) ∨ ⟨b,a⟩ ∈ StrictVersion(r)
 ⟨*proof*⟩

A trichotomy law for the strict version of a total and antisymmetric relation. It is kind of interesting that one does not need the full linear order for this.

lemma strict˙ans˙tot˙trich:
 assumes A1: antisym(r) and A2: r –is total on″ X
 and A3: a∈X b∈X
 and A4: s = StrictVersion(r)
 shows Exactly˙1˙of˙3˙holds(⟨a,b⟩ ∈ s, a=b,⟨b,a⟩ ∈ s)
⟨*proof*⟩

A trichotomy law for linear order. This is a special case of strict˙ans˙tot˙trich.

corollary strict˙lin˙trich: assumes A1: IsLinOrder(X,r) and
 A2: a∈X b∈X and
 A3: s = StrictVersion(r)
 shows Exactly˙1˙of˙3˙holds(⟨a,b⟩ ∈ s, a=b,⟨b,a⟩ ∈ s)
 ⟨*proof*⟩

For an antisymmetric relation if a pair is in relation then the reversed pair is not in the strict version of the relation.

lemma geq˙impl˙not˙less:
 assumes A1: antisym(r) and A2: ⟨a,b⟩ ∈ r
 shows ⟨b,a⟩ ∉ StrictVersion(r)

$\langle proof \rangle$

If an antisymmetric relation is transitive, then the strict version is also transitive, an explicit version strict˙of˙transB below.

lemma strict˙of˙transA:
 assumes A1: trans(r) and A2: antisym(r) and
 A3: s= StrictVersion(r) and A4: ⟨a,b⟩ ∈ s ⟨b,c⟩ ∈ s
 shows ⟨a,c⟩ ∈ s
$\langle proof \rangle$

If an antisymmetric relation is transitive, then the strict version is also transitive.

lemma strict˙of˙transB:
 assumes A1: trans(r) and A2: antisym(r)
 shows trans(StrictVersion(r))
$\langle proof \rangle$

The next lemma provides a condition that is satisfied by the strict version of a relation if the original relation is a complete linear order.

lemma strict˙of˙compl:
 assumes A1: r ⊆ X×X and A2: IsLinOrder(X,r) and
 A3: r –is complete˝ and
 A4: A⊆X A≠0 and A5: s = StrictVersion(r) and
 A6: ∃u∈X. ∀y∈A. ⟨y,u⟩ ∈ s
 shows
 ∃x∈X. (∀y∈A. ⟨x,y⟩ ∉ s) ∧ (∀y∈X. ⟨y,x⟩ ∈ s ⟶ (∃z∈A. ⟨y,z⟩ ∈ s))
$\langle proof \rangle$

Strict version of a relation on a set is a relation on that set.

lemma strict˙ver˙rel: assumes A1: r ⊆ A×A
 shows StrictVersion(r) ⊆ A×A
 $\langle proof \rangle$

end

8 Functions - introduction

theory func1 imports ZF.func Fol1 ZF1

begin

This theory covers basic properties of function spaces. A set of functions with domain X and values in the set Y is denoted in Isabelle as $X \to Y$. It just happens that the colon ":" is a synonym of the set membership symbol $\in$ in Isabelle/ZF so we can write $f : X \to Y$ instead of $f \in X \to Y$. This is the only case that we use the colon instead of the regular set membership symbol.

8.1 Properties of functions, function spaces and (inverse) images.

Functions in ZF are sets of pairs. This means that if $f : X \to Y$ then $f \subseteq X \times Y$. This section is mostly about consequences of this understanding of the notion of function.

We define the notion of function that preserves a collection here. Given two collection of sets a function preserves the collections if the inverse image of sets in one collection belongs to the second one. This notion does not have a name in romantic math. It is used to define continuous functions in Topology_ZF_2 theory. We define it here so that we can use it for other purposes, like defining measurable functions. Recall that f-(A) means the inverse image of the set A.

definition
  PresColl(f,S,T) ≡ ∀ A∈T. f-(A)∈S

A definition that allows to get the first factor of the domain of a binary function $f : X \times Y \to Z$.

definition
  fstdom(f) ≡ domain(domain(f))

If a function maps A into another set, then A is the domain of the function.

lemma func1_1_L1: assumes f:A→C shows domain(f) = A
  ⟨*proof*⟩

Standard Isabelle defines a function(f) predicate. The next lemma shows that our functions satisfy that predicate. It is a special version of Isabelle's fun_is_function.

lemma fun_is_fun: assumes f:X→Y shows function(f)
  ⟨*proof*⟩

A lemma explains what fstdom is for.

lemma fstdomdef: assumes A1: f: X×Y → Z and A2: Y≠0
  shows fstdom(f) = X
⟨*proof*⟩

A version of the Pi_type lemma from the standard Isabelle/ZF library.

lemma func1_1_L1A: assumes A1: f:X→Y and A2: ∀x∈X. f(x) ∈ Z
  shows f:X→Z
⟨*proof*⟩

A variant of func1_1_L1A.

lemma func1_1_L1B: assumes A1: f:X→Y and A2: Y⊆Z
  shows f:X→Z
⟨*proof*⟩

There is a value for each argument.

lemma func1˙1˙L2: assumes A1: f:X→Y x∈X
shows ∃y∈Y. ⟨x,y⟩ ∈ f
⟨*proof*⟩

The inverse image is the image of converse. True for relations as well.

lemma vimage˙converse: shows r-(A) = converse(r)(A)
⟨*proof*⟩

The image is the inverse image of converse.

lemma image˙converse: shows converse(r)-(A) = r(A)
⟨*proof*⟩

The inverse image by a composition is the composition of inverse images.

lemma vimage˙comp: shows (r O s)-(A) = s-(r-(A))
⟨*proof*⟩

A version of vimage˙comp for three functions.

lemma vimage˙comp3: shows (r O s O t)-(A) = t-(s-(r-(A)))
⟨*proof*⟩

Inverse image of any set is contained in the domain.

lemma func1˙1˙L3: assumes A1: f:X→Y shows f-(D) ⊆ X
⟨*proof*⟩

The inverse image of the range is the domain.

lemma func1˙1˙L4: assumes f:X→Y shows f-(Y) = X
⟨*proof*⟩

The arguments belongs to the domain and values to the range.

lemma func1˙1˙L5:
assumes A1: ⟨ x,y⟩ ∈ f and A2: f:X→Y
shows x∈X ∧ y∈Y
⟨*proof*⟩

Function is a subset of cartesian product.

lemma fun˙subset˙prod: assumes A1: f:X→Y shows f ⊆ X×Y
⟨*proof*⟩

The (argument, value) pair belongs to the graph of the function.

lemma func1˙1˙L5A:
assumes A1: f:X→Y x∈X y = f(x)
shows ⟨x,y⟩ ∈ f y ∈ range(f)
⟨*proof*⟩

The next theorem illustrates the meaning of the concept of function in ZF.

theorem fun˙is˙set˙of˙pairs: assumes A1: f:X→Y
shows f = –⟨x, f(x)⟩. x ∈ X″

⟨*proof*⟩

The range of function that maps X into Y is contained in Y.

lemma func1˙1˙L5B:
 assumes A1: f:X→Y shows range(f) ⊆ Y
⟨*proof*⟩

The image of any set is contained in the range.

lemma func1˙1˙L6: assumes A1: f:X→Y
 shows f(B) ⊆ range(f) and f(B) ⊆ Y
⟨*proof*⟩

The inverse image of any set is contained in the domain.

lemma func1˙1˙L6A: assumes A1: f:X→Y shows f-(A)⊆X
⟨*proof*⟩

Image of a greater set is greater.

lemma func1˙1˙L8: assumes A1: A⊆B shows f(A)⊆ f(B)
 ⟨*proof*⟩

A set is contained in the the inverse image of its image. There is similar theorem in equalities.thy (function˙image˙vimage) which shows that the image of inverse image of a set is contained in the set.

lemma func1˙1˙L9: assumes A1: f:X→Y and A2: A⊆X
 shows A ⊆ f-(f(A))
⟨*proof*⟩

The inverse image of the image of the domain is the domain.

lemma inv˙im˙dom: assumes A1: f:X→Y shows f-(f(X)) = X
⟨*proof*⟩

A technical lemma needed to make the func1˙1˙L11 proof more clear.

lemma func1˙1˙L10:
 assumes A1: f ⊆ X×Y and A2: ∃!y. (y∈Y ∧ ⟨x,y⟩ ∈ f)
 shows ∃!y. ⟨x,y⟩ ∈ f
⟨*proof*⟩

If $f \subseteq X \times Y$ and for every $x \in X$ there is exactly one $y \in Y$ such that $(x, y) \in f$ then f maps X to Y.

lemma func1˙1˙L11:
 assumes f ⊆ X×Y and ∀x∈X. ∃!y. y∈Y ∧ ⟨x,y⟩ ∈ f
 shows f: X→Y ⟨*proof*⟩

A set defined by a lambda-type expression is a fuction. There is a similar lemma in func.thy, but I had problems with lambda expressions syntax so I could not apply it. This lemma is a workaround for this. Besides, lambda expressions are not readable.

lemma func1˙1˙L11A: assumes A1: $\forall x \in X.\ b(x) \in Y$
shows $–\langle x,y\rangle \in X\times Y.\ b(x) = y''$: $X\to Y$
⟨*proof*⟩

The next lemma will replace func1˙1˙L11A one day.

lemma ZF˙fun˙from˙total: assumes A1: $\forall x \in X.\ b(x) \in Y$
shows $–\langle x,b(x)\rangle.\ x\in X''$: $X\to Y$
⟨*proof*⟩

The value of a function defined by a meta-function is this meta-function.

lemma func1˙1˙L11B:
assumes A1: $f:X\to Y$ $x\in X$
and A2: $f = –\langle x,y\rangle \in X\times Y.\ b(x) = y''$
shows $f(x) = b(x)$
⟨*proof*⟩

The next lemma will replace func1˙1˙L11B one day.

lemma ZF˙fun˙from˙tot˙val:
assumes A1: $f:X\to Y$ $x\in X$
and A2: $f = –\langle x,b(x)\rangle.\ x\in X''$
shows $f(x) = b(x)$
⟨*proof*⟩

Identical meaning as ZF˙fun˙from˙tot˙val, but phrased a bit differently.

lemma ZF˙fun˙from˙tot˙val0:
assumes $f:X\to Y$ and $f = –\langle x,b(x)\rangle.\ x\in X''$
shows $\forall x\in X.\ f(x) = b(x)$
⟨*proof*⟩

Another way of expressing that lambda expression is a function.

lemma lam˙is˙fun˙range: assumes $f=–\langle x,g(x)\rangle.\ x\in X''$
shows $f:X\to range(f)$
⟨*proof*⟩

Yet another way of expressing value of a function.

lemma ZF˙fun˙from˙tot˙val1:
assumes $x\in X$ shows $–\langle x,b(x)\rangle.\ x\in X''(x)=b(x)$
⟨*proof*⟩

We can extend a function by specifying its values on a set disjoint with the domain.

lemma func1˙1˙L11C: assumes A1: $f:X\to Y$ and A2: $\forall x\in A.\ b(x)\in B$
and A3: $X\cap A = 0$ and Dg: $g = f \cup –\langle x,b(x)\rangle.\ x\in A''$
shows
$g : X\cup A \to Y\cup B$
$\forall x\in X.\ g(x) = f(x)$
$\forall x\in A.\ g(x) = b(x)$

⟨*proof*⟩

We can extend a function by specifying its value at a point that does not belong to the domain.

lemma func1˙1˙L11D: assumes A1: f:X→Y and A2: a∉X
  and Dg: g = f ∪ –⟨a,b⟩″
  shows
  g : X∪–a″ → Y∪–b″
  ∀x∈X. g(x) = f(x)
  g(a) = b
⟨*proof*⟩

A technical lemma about extending a function both by defining on a set disjoint with the domain and on a point that does not belong to any of those sets.

lemma func1˙1˙L11E:
  assumes A1: f:X→Y and
  A2: ∀x∈A. b(x)∈B and
  A3: X∩A = 0 and A4: a∉ X∪A
  and Dg: g = f ∪ –⟨x,b(x)⟩. x∈A″ ∪ –⟨a,c⟩″
  shows
  g : X∪A∪–a″ → Y∪B∪–c″
  ∀x∈X. g(x) = f(x)
  ∀x∈A. g(x) = b(x)
  g(a) = c
⟨*proof*⟩

A way of defining a function on a union of two possibly overlapping sets. We decompose the union into two differences and the intersection and define a function separately on each part.

lemma fun˙union˙overlap: assumes ∀x∈A∩B. h(x) ∈ Y ∀x∈A-B. f(x) ∈ Y ∀x∈B-A. g(x) ∈ Y
  shows –⟨x,if x∈A-B then f(x) else if x∈B-A then g(x) else h(x)⟩. x ∈ A∪B″: A∪B → Y
⟨*proof*⟩

Inverse image of intersection is the intersection of inverse images.

lemma invim˙inter˙inter˙invim: assumes f:X→Y
  shows f-(A∩B) = f-(A) ∩ f-(B)
  ⟨*proof*⟩

The inverse image of an intersection of a nonempty collection of sets is the intersection of the inverse images. This generalizes invim˙inter˙inter˙invim which is proven for the case of two sets.

lemma func1˙1˙L12:
  assumes A1: B ⊆ Pow(Y) and A2: B≠0 and A3: f:X→Y
  shows f-(⋂B) = (⋂U∈B. f-(U))

⟨*proof*⟩

The inverse image of a set does not change when we intersect the set with the image of the domain.

lemma inv˙im˙inter˙im: assumes f:X→Y
shows f-(A ∩ f(X)) = f-(A)
⟨*proof*⟩

If the inverse image of a set is not empty, then the set is not empty. Proof by contradiction.

lemma func1˙1˙L13: assumes A1:f-(A) ≠ 0 shows A≠0
⟨*proof*⟩

If the image of a set is not empty, then the set is not empty. Proof by contradiction.

lemma func1˙1˙L13A: assumes A1: f(A)≠0 shows A≠0
⟨*proof*⟩

What is the inverse image of a singleton?

lemma func1˙1˙L14: assumes f∈X→Y
shows f-(–y″) = –x∈X. f(x) = y″
⟨*proof*⟩

A lemma that can be used instead fun˙extension˙iff to show that two functions are equal

lemma func˙eq: assumes f: X→Y g: X→Z
and ∀x∈X. f(x) = g(x)
shows f = g ⟨*proof*⟩

Function defined on a singleton is a single pair.

lemma func˙singleton˙pair: assumes A1: f : –a″→X
shows f = –⟨a, f(a)⟩″
⟨*proof*⟩

A single pair is a function on a singleton. This is similar to singleton˙fun from standard Isabelle/ZF.

lemma pair˙func˙singleton: assumes A1: y ∈ Y
shows –⟨x,y⟩″ : –x″ → Y
⟨*proof*⟩

The value of a pair on the first element is the second one.

lemma pair˙val: shows –⟨x,y⟩″(x) = y
⟨*proof*⟩

A more familiar definition of inverse image.

lemma func1˙1˙L15: assumes A1: f:X→Y

shows f-(A) = –x∈X. f(x) ∈ A″
⟨*proof*⟩

A more familiar definition of image.

lemma func˙imagedef: assumes A1: f:X→Y and A2: A⊆X
shows f(A) = –f(x). x ∈ A″
⟨*proof*⟩

The image of a set contained in domain under identity is the same set.

lemma image˙id˙same: assumes A⊆X shows id(X)(A) = A
⟨*proof*⟩

The inverse image of a set contained in domain under identity is the same set.

lemma vimage˙id˙same: assumes A⊆X shows id(X)-(A) = A
⟨*proof*⟩

What is the image of a singleton?

lemma singleton˙image:
assumes f∈X→Y and x∈X
shows f–x″ = –f(x)″
⟨*proof*⟩

If an element of the domain of a function belongs to a set, then its value belongs to the imgage of that set.

lemma func1˙1˙L15D: assumes f:X→Y x∈A A⊆X
shows f(x) ∈ f(A)
⟨*proof*⟩

Range is the image of the domain. Isabelle/ZF defines range(f) as domain(converse(f)), and that's why we have something to prove here.

lemma range˙image˙domain:
assumes A1: f:X→Y shows f(X) = range(f)
⟨*proof*⟩

The difference of images is contained in the image of difference.

lemma diff˙image˙diff: assumes A1: f: X→Y and A2: A⊆X
shows f(X) - f(A) ⊆ f(X-A)
⟨*proof*⟩

The image of an intersection is contained in the intersection of the images.

lemma image˙of˙Inter: assumes A1: f:X→Y and
A2: I≠0 and A3: ∀i∈I. P(i) ⊆ X
shows f(⋂i∈I. P(i)) ⊆ (⋂i∈I. f(P(i)))
⟨*proof*⟩

The image of union is the union of images.

lemma image˙of˙Union: assumes A1: f:X→Y and A2: ∀A∈M. A⊆X
shows f(⋃M) = ⋃–f(A). A∈M″
⟨*proof*⟩

The image of a nonempty subset of domain is nonempty.

lemma func1˙1˙L15A:
assumes A1: f: X→Y and A2: A⊆X and A3: A≠0
shows f(A) ≠ 0
⟨*proof*⟩

The next lemma allows to prove statements about the values in the domain of a function given a statement about values in the range.

lemma func1˙1˙L15B:
assumes f:X→Y and A⊆X and ∀y∈f(A). P(y)
shows ∀x∈A. P(f(x))
⟨*proof*⟩

An image of an image is the image of a composition.

lemma func1˙1˙L15C: assumes A1: f:X→Y and A2: g:Y→Z
and A3: A⊆X
shows
g(f(A)) = –g(f(x)). x∈A″
g(f(A)) = (g O f)(A)
⟨*proof*⟩

What is the image of a set defined by a meta-fuction?

lemma func1˙1˙L17:
assumes A1: f ∈ X→Y and A2: ∀x∈A. b(x) ∈ X
shows f(–b(x). x∈A″) = –f(b(x)). x∈A″
⟨*proof*⟩

What are the values of composition of three functions?

lemma func1˙1˙L18: assumes A1: f:A→B g:B→C h:C→D
and A2: x∈A
shows
(h O g O f)(x) ∈ D
(h O g O f)(x) = h(g(f(x)))
⟨*proof*⟩

A composition of functions is a function. This is a slight generalization of standard Isabelle's comp˙fun

lemma comp˙fun˙subset:
assumes A1: g:A→B and A2: f:C→D and A3: B ⊆ C
shows f O g : A → D
⟨*proof*⟩

This lemma supersedes the lemma comp˙eq˙id˙iff in Isabelle/ZF. Contributed by Victor Porton.

lemma comp˙eq˙id˙iff1: assumes A1: g: B→A and A2: f: A→C
  shows (∀y∈B. f(g(y)) = y) ⟷ f O g = id(B)
⟨proof⟩

A lemma about a value of a function that is a union of some collection of functions.

lemma fun˙Union˙apply: assumes A1: ⋃F : X→Y and
  A2: f∈F and A3: f:A→B and A4: x∈A
  shows (⋃F)(x) = f(x)
⟨proof⟩

8.2 Functions restricted to a set

Standard Isabelle/ZF defines the notion restrict(f,A) of to mean a function (or relation) f restricted to a set. This means that if f is a function defined on X and A is a subset of X then restrict(f,A) is a function whith the same values as f, but whose domain is A.

What is the inverse image of a set under a restricted fuction?

lemma func1˙2˙L1: assumes A1: f:X→Y and A2: B⊆X
  shows restrict(f,B)-(A) = f-(A) ∩ B
⟨proof⟩

A criterion for when one function is a restriction of another. The lemma below provides a result useful in the actual proof of the criterion and applications.

lemma func1˙2˙L2:
  assumes A1: f:X→Y and A2: g ∈ A→Z
  and A3: A⊆X and A4: f ∩ A×Z = g
  shows ∀x∈A. g(x) = f(x)
⟨proof⟩

Here is the actual criterion.

lemma func1˙2˙L3:
  assumes A1: f:X→Y and A2: g:A→Z
  and A3: A⊆X and A4: f ∩ A×Z = g
  shows g = restrict(f,A)
⟨proof⟩

Which function space a restricted function belongs to?

lemma func1˙2˙L4:
  assumes A1: f:X→Y and A2: A⊆X and A3: ∀x∈A. f(x) ∈ Z
  shows restrict(f,A) : A→Z
⟨proof⟩

A simpler case of func1˙2˙L4, where the range of the original and restricted function are the same.

corollary restrict˙fun: assumes A1: f:X→Y and A2: A⊆X
  shows restrict(f,A) : A → Y
⟨proof⟩

A composition of two functions is the same as composition with a restriction.

lemma comp˙restrict:
  assumes A1: f : A→B and A2: g : X → C and A3: B⊆X
  shows g O f = restrict(g,B) O f
⟨proof⟩

A way to look at restriction. Contributed by Victor Porton.

lemma right˙comp˙id˙any: shows r O id(C) = restrict(r,C)
  ⟨proof⟩

8.3 Constant functions

Constant functions are trivial, but still we need to prove some properties to shorten proofs.

We define constant$(=c)$ functions on a set X in a natural way as ConstantFunction(X,c).

definition
  ConstantFunction(X,c) ≡ X×–c˝

Constant function belongs to the function space.

lemma func1˙3˙L1:
  assumes A1: c∈Y shows ConstantFunction(X,c) : X→Y
⟨proof⟩

Constant function is equal to the constant on its domain.

lemma func1˙3˙L2: assumes A1: x∈X
  shows ConstantFunction(X,c)(x) = c
⟨proof⟩

8.4 Injections, surjections, bijections etc.

In this section we prove the properties of the spaces of injections, surjections and bijections that we can't find in the standard Isabelle's Perm.thy.

For injections the image a difference of two sets is the difference of images

lemma inj˙image˙dif:
  assumes A1: f ∈ inj(A,B) and A2: C ⊆ A
  shows f(A-C) = f(A) - f(C)
⟨proof⟩

For injections the image of intersection is the intersection of images.

lemma inj˙image˙inter: assumes A1: f ∈ inj(X,Y) and A2: A⊆X B⊆X
  shows f(A∩B) = f(A) ∩ f(B)

⟨*proof*⟩

For surjection from A to B the image of the domain is B.

lemma surj˙range˙image˙domain: assumes A1: f ∈ surj(A,B)
  shows f(A) = B
⟨*proof*⟩

For injections the inverse image of an image is the same set.

lemma inj˙vimage˙image: assumes f ∈ inj(X,Y) and A⊆X
  shows f-(f(A)) = A
⟨*proof*⟩

For surjections the image of an inverse image is the same set.

lemma surj˙image˙vimage: assumes A1: f ∈ surj(X,Y) and A2: A⊆Y
  shows f(f-(A)) = A
⟨*proof*⟩

A lemma about how a surjection maps collections of subsets in domain and rangge.

lemma surj˙subsets: assumes A1: f ∈ surj(X,Y) and A2: B ⊆ Pow(Y)
  shows – f(U). U ∈ –f-(V). V∈B˝ ˝ = B
⟨*proof*⟩

Restriction of an bijection to a set without a point is a a bijection.

lemma bij˙restrict˙rem:
  assumes A1: f ∈ bij(A,B) and A2: a∈A
  shows restrict(f, A-–a˝) ∈ bij(A-–a˝, B-–f(a)˝)
⟨*proof*⟩

The domain of a bijection between X and Y is X.

lemma domain˙of˙bij:
  assumes A1: f ∈ bij(X,Y) shows domain(f) = X
⟨*proof*⟩

The value of the inverse of an injection on a point of the image of a set belongs to that set.

lemma inj˙inv˙back˙in˙set:
  assumes A1: f ∈ inj(A,B) and A2: C⊆A and A3: y ∈ f(C)
  shows
  converse(f)(y) ∈ C
  f(converse(f)(y)) = y
⟨*proof*⟩

For injections if a value at a point belongs to the image of a set, then the point belongs to the set.

lemma inj˙point˙of˙image:
  assumes A1: f ∈ inj(A,B) and A2: C⊆A and

A3: x∈A and A4: f(x) ∈ f(C)
shows x ∈ C
⟨*proof*⟩

For injections the image of intersection is the intersection of images.

lemma inj_image_of_Inter: assumes A1: f ∈ inj(A,B) and
A2: I≠0 and A3: ∀i∈I. P(i) ⊆ A
shows f(⋂i∈I. P(i)) = (⋂i∈I. f(P(i)))
⟨*proof*⟩

An injection is injective onto its range. Suggested by Victor Porton.

lemma inj_inj_range: assumes f ∈ inj(A,B)
shows f ∈ inj(A,range(f))
⟨*proof*⟩

An injection is a bijection on its range. Suggested by Victor Porton.

lemma inj_bij_range: assumes f ∈ inj(A,B)
shows f ∈ bij(A,range(f))
⟨*proof*⟩

A lemma about extending a surjection by one point.

lemma surj_extend_point:
assumes A1: f ∈ surj(X,Y) and A2: a∉X and
A3: g = f ∪ {⟨a,b⟩}
shows g ∈ surj(X∪{a},Y∪{b})
⟨*proof*⟩

A lemma about extending an injection by one point. Essentially the same as standard Isabelle's inj_extend.

lemma inj_extend_point: assumes f ∈ inj(X,Y) a∉X b∉Y
shows (f ∪ {⟨a,b⟩}) ∈ inj(X∪{a},Y∪{b})
⟨*proof*⟩

A lemma about extending a bijection by one point.

lemma bij_extend_point: assumes f ∈ bij(X,Y) a∉X b∉Y
shows (f ∪ {⟨a,b⟩}) ∈ bij(X∪{a},Y∪{b})
⟨*proof*⟩

A quite general form of the $a^{-1}b = 1$ implies $a = b$ law.

lemma comp_inv_id_eq:
assumes A1: converse(b) O a = id(A) and
A2: a ⊆ A×B b ∈ surj(A,B)
shows a = b
⟨*proof*⟩

A special case of comp_inv_id_eq - the $a^{-1}b = 1$ implies $a = b$ law for bijections.

lemma comp_inv_id_eq_bij:

assumes A1: $a \in bij(A,B)$ $b \in bij(A,B)$ and
A2: $converse(b)\ O\ a = id(A)$
shows $a = b$
⟨*proof*⟩

Converse of a converse of a bijection is the same bijection. This is a special case of converse˙converse from standard Isabelle's equalities theory where it is proved for relations.

lemma bij˙converse˙converse: assumes $a \in bij(A,B)$
shows $converse(converse(a)) = a$
⟨*proof*⟩

If a composition of bijections is identity, then one is the inverse of the other.

lemma comp˙id˙conv: assumes A1: $a \in bij(A,B)$ $b \in bij(B,A)$ and
A2: $b\ O\ a = id(A)$
shows $a = converse(b)$ and $b = converse(a)$
⟨*proof*⟩

A version of comp˙id˙conv with weaker assumptions.

lemma comp˙conv˙id: assumes A1: $a \in bij(A,B)$ and A2: $b:B\rightarrow A$ and
A3: $\forall x\in A.\ b(a(x)) = x$
shows $b \in bij(B,A)$ and $a = converse(b)$ and $b = converse(a)$
⟨*proof*⟩

For a surjection the union if images of singletons is the whole range.

lemma surj˙singleton˙image: assumes A1: $f \in surj(X,Y)$
shows $(\bigcup x\in X.\ \{f(x)\}) = Y$
⟨*proof*⟩

8.5 Functions of two variables

In this section we consider functions whose domain is a cartesian product of two sets. Such functions are called functions of two variables (although really in ZF all functions admit only one argument). For every function of two variables we can define families of functions of one variable by fixing the other variable. This section establishes basic definitions and results for this concept.

We can create functions of two variables by combining functions of one variable.

lemma cart˙prod˙fun: assumes $f_1:X_1\rightarrow Y_1$ $f_2:X_2\rightarrow Y_2$ and
$g = \{\langle p,\langle f_1(fst(p)),f_2(snd(p))\rangle\rangle.\ p \in X_1\times X_2\}$
shows $g: X_1\times X_2 \rightarrow Y_1\times Y_2$ ⟨*proof*⟩

A reformulation of cart˙prod˙fun above in a sligtly different notation.

lemma prod˙fun:
assumes $f:X_1\rightarrow X_2$ $g:X_3\rightarrow X_4$

shows $\{\langle\langle x,y\rangle,\langle fx,gy\rangle\rangle.\ \langle x,y\rangle \in X_1 \times X_3\}:X_1 \times X_3 \to X_2 \times X_4$
⟨*proof*⟩

Product of two surjections is a surjection.

theorem prod_functions_surj:
assumes $f \in surj(A,B)$ $g \in surj(C,D)$
shows $\{\langle\langle a1,a2\rangle,\langle fa1,ga2\rangle\rangle.\langle a1,a2\rangle \in A \times C\} \in surj(A \times C,B \times D)$
⟨*proof*⟩

For a function of two variables created from functions of one variable as in cart_prod_fun above, the inverse image of a cartesian product of sets is the cartesian product of inverse images.

lemma cart_prod_fun_vimage: assumes $f_1:X_1 \to Y_1$ $f_2:X_2 \to Y_2$ and
$g = \{\langle p,\langle f_1(fst(p)),f_2(snd(p))\rangle\rangle.\ p \in X_1 \times X_2\}$
shows $g\text{-}(A_1 \times A_2) = f_1\text{-}(A_1) \times f_2\text{-}(A_2)$
⟨*proof*⟩

For a function of two variables defined on $X \times Y$, if we fix an $x \in X$ we obtain a function on Y. Note that if domain(f) is $X \times Y$, range(domain(f)) extracts Y from $X \times Y$.

definition
Fix1stVar(f,x) $\equiv \{\langle y,f\langle x,y\rangle\rangle.\ y \in range(domain(f))\}$

For every $y \in Y$ we can fix the second variable in a binary function $f : X \times Y \to Z$ to get a function on X.

definition
Fix2ndVar(f,y) $\equiv \{\langle x,f\langle x,y\rangle\rangle.\ x \in domain(domain(f))\}$

We defined Fix1stVar and Fix2ndVar so that the domain of the function is not listed in the arguments, but is recovered from the function. The next lemma is a technical fact that makes it easier to use this definition.

lemma fix_var_fun_domain: assumes A1: $f : X \times Y \to Z$
shows
$x \in X \longrightarrow$ Fix1stVar(f,x) $= \{\langle y,f\langle x,y\rangle\rangle.\ y \in Y\}$
$y \in Y \longrightarrow$ Fix2ndVar(f,y) $= \{\langle x,f\langle x,y\rangle\rangle.\ x \in X\}$
⟨*proof*⟩

If we fix the first variable, we get a function of the second variable.

lemma fix_1st_var_fun: assumes A1: $f : X \times Y \to Z$ and A2: $x \in X$
shows Fix1stVar(f,x) : $Y \to Z$
⟨*proof*⟩

If we fix the second variable, we get a function of the first variable.

lemma fix_2nd_var_fun: assumes A1: $f : X \times Y \to Z$ and A2: $y \in Y$
shows Fix2ndVar(f,y) : $X \to Z$
⟨*proof*⟩

What is the value of Fix1stVar(f,x) at $y \in Y$ and the value of Fix2ndVar(f,y) at $x \in X$"?

lemma fix˙var˙val:
  assumes A1: f : X×Y → Z and A2: x∈X y∈Y
  shows
  Fix1stVar(f,x)(y) = f⟨x,y⟩
  Fix2ndVar(f,y)(x) = f⟨x,y⟩
⟨*proof*⟩

Fixing the second variable commutes with restrictig the domain.

lemma fix˙2nd˙var˙restr˙comm:
  assumes A1: f : X×Y → Z and A2: y∈Y and A3: X_1 ⊆ X
  shows Fix2ndVar(restrict(f,X_1×Y),y) = restrict(Fix2ndVar(f,y),X_1)
⟨*proof*⟩

The next lemma expresses the inverse image of a set by function with fixed first variable in terms of the original function.

lemma fix˙1st˙var˙vimage:
  assumes A1: f : X×Y → Z and A2: x∈X
  shows Fix1stVar(f,x)-(A) = –y∈Y. ⟨x,y⟩ ∈ f-(A)″
⟨*proof*⟩

The next lemma expresses the inverse image of a set by function with fixed second variable in terms of the original function.

lemma fix˙2nd˙var˙vimage:
  assumes A1: f : X×Y → Z and A2: y∈Y
  shows Fix2ndVar(f,y)-(A) = –x∈X. ⟨x,y⟩ ∈ f-(A)″
⟨*proof*⟩

end

9 Semilattices and Lattices

theory Lattice˙ZF imports Order˙ZF˙1a func1

begin

Lattices can be introduced in algebraic way as commutative idempotent $(x \cdot x = x)$ semigroups or as partial orders with some additional properties. These two approaches are equivalent. In this theory we will use the order-theoretic approach.

9.1 Semilattices

We start with a relation r which is a partial order on a set L. Such situation is defined in Order˙ZF as the predicate IsPartOrder(L,r).

A partially ordered (L, r) set is a join-semilattice if each two-element subset of L has a supremum (i.e. the least upper bound).

definition
IsJoinSemilattice(L,r) ≡
r⊆L×L ∧ IsPartOrder(L,r) ∧ (∀x∈L. ∀y∈L. HasAsupremum(r,–x,y˝))

A partially ordered (L, r) set is a meet-semilattice if each two-element subset of L has an infimum (i.e. the greatest lower bound).

definition
IsMeetSemilattice(L,r) ≡
r⊆L×L ∧ IsPartOrder(L,r) ∧ (∀x∈L. ∀y∈L. HasAnInfimum(r,–x,y˝))

A partially ordered (L, r) set is a lattice if it is both join and meet-semilattice, i.e. if every two element set has a supremum (least upper bound) and infimum (greatest lower bound).

definition
IsAlattice (infixl –is a lattice on˝ 90) where
r –is a lattice on˝ L ≡ IsJoinSemilattice(L,r) ∧ IsMeetSemilattice(L,r)

Join is a binary operation whose value on a pair $\langle x, y\rangle$ is defined as the supremum of the set $\{x, y\}$.

definition
Join(L,r) ≡ –⟨p,Supremum(r,–fst(p),snd(p)˝)⟩ . p ∈ L×L˝

Meet is a binary operation whose value on a pair $\langle x, y\rangle$ is defined as the infimum of the set $\{x, y\}$.

definition
Meet(L,r) ≡ –⟨p,Infimum(r,–fst(p),snd(p)˝)⟩ . p ∈ L×L˝

In a join-semilattice join is indeed a binary operation.

lemma join˙is˙binop: assumes IsJoinSemilattice(L,r)
shows Join(L,r) : L×L → L
⟨*proof*⟩

The value of Join(L,r) on a pair $\langle x, y\rangle$ is the supremum of the set $\{x, y\}$, hence its is greater or equal than both.

lemma join˙val:
assumes IsJoinSemilattice(L,r) x∈L y∈L
defines j ≡ Join(L,r)⟨x,y⟩
shows j∈L j = Supremum(r,–x,y˝) ⟨x,j⟩ ∈ r ⟨y,j⟩ ∈ r
⟨*proof*⟩

In a meet-semilattice meet is indeed a binary operation.

lemma meet˙is˙binop: assumes IsMeetSemilattice(L,r)
shows Meet(L,r) : L×L → L
⟨*proof*⟩

The value of Meet(L,r) on a pair $\langle x, y\rangle$ is the infimum of the set $\{x, y\}$, hence is less or equal than both.

lemma meet˙val:
 assumes IsMeetSemilattice(L,r) x∈L y∈L
 defines m ≡ Meet(L,r)⟨x,y⟩
 shows m∈L m = Infimum(r,–x,y″) ⟨m,x⟩ ∈ r ⟨m,y⟩ ∈ r
⟨*proof*⟩

The next locale defines a a notation for join-semilattice. We will use the ⊔ symbol rather than more common ∨ to avoid confusion with logical ”or”.

locale join˙semilatt =
 fixes L
 fixes r
 assumes joinLatt: IsJoinSemilattice(L,r)
 fixes join (infixl ⊔ 71)
 defines join˙def [simp]: x ⊔ y ≡ Join(L,r)⟨x,y⟩
 fixes sup (sup ˙)
 defines sup˙def [simp]: sup A ≡ Supremum(r,A)

Join of the elements of the lattice is in the lattice.

lemma (in join˙semilatt) join˙props: assumes x∈L y∈L
 shows x⊔y ∈ L and x⊔y = sup –x,y″
⟨*proof*⟩

Join is associative.

lemma (in join˙semilatt) join˙assoc: assumes x∈L y∈L z∈L
 shows x⊔(y⊔z) = x⊔y⊔z
⟨*proof*⟩

Join is idempotent.

lemma (in join˙semilatt) join˙idempotent: assumes x∈L shows x⊔x = x
 ⟨*proof*⟩

The meet˙semilatt locale is the dual of the join-semilattice locale defined above. We will use the ⊓ symbol to denote join, giving it ab bit higher precedence.

locale meet˙semilatt =
 fixes L
 fixes r
 assumes meetLatt: IsMeetSemilattice(L,r)
 fixes join (infixl ⊓ 72)
 defines join˙def [simp]: x ⊓ y ≡ Meet(L,r)⟨x,y⟩
 fixes sup (inf ˙)
 defines sup˙def [simp]: inf A ≡ Infimum(r,A)

Meet of the elements of the lattice is in the lattice.

lemma (in meet˙semilatt) meet˙props: assumes x∈L y∈L

shows x⊓y ∈ L and x⊓y = inf –x,y˝
⟨*proof*⟩

Meet is associative.

lemma (in meet˙semilatt) meet˙assoc: assumes x∈L y∈L z∈L
shows x⊓(y⊓z) = x⊓y⊓z
⟨*proof*⟩

Meet is idempotent.

lemma (in meet˙semilatt) meet˙idempotent: assumes x∈L shows x⊓x = x
⟨*proof*⟩

end

10 Order on natural numbers

theory NatOrder˙ZF imports Nat˙ZF˙IML Order˙ZF

begin

This theory proves that ≤ is a linear order on ℕ. ≤ is defined in Isabelle's Nat theory, and linear order is defined in Order˙ZF theory. Contributed by Seo Sanghyeon.

10.1 Order on natural numbers

This is the only section in this theory.

To prove that ≤ is a total order, we use a result on ordinals.

lemma NatOrder˙ZF˙1˙L1:
assumes a∈nat and b∈nat
shows a ≤ b ∨ b ≤ a
⟨*proof*⟩

≤ is antisymmetric, transitive, total, and linear. Proofs by rewrite using definitions.

lemma NatOrder˙ZF˙1˙L2:
shows
antisym(Le)
trans(Le)
Le –is total on˝ nat
IsLinOrder(nat,Le)
⟨*proof*⟩

The order on natural numbers is linear on every natural number. Recall that each natural number is a subset of the set of all natural numbers (as well as a member).

lemma natord˙lin˙on˙each˙nat:
  assumes A1: n ∈ nat shows IsLinOrder(n,Le)
⟨*proof*⟩

end

11 Binary operations

theory func˙ZF imports func1

begin

In this theory we consider properties of functions that are binary operations, that is they map $X \times X$ into X.

11.1 Lifting operations to a function space

It happens quite often that we have a binary operation on some set and we need a similar operation that is defined for functions on that set. For example once we know how to add real numbers we also know how to add real-valued functions: for $f, g : X \to \mathbf{R}$ we define $(f + g)(x) = f(x) + g(x)$. Note that formally the + means something different on the left hand side of this equality than on the right hand side. This section aims at formalizing this process. We will call it "lifting to a function space", if you have a suggestion for a better name, please let me know.

Since we are writing in generic set notation, the definition below is a bit complicated. Here it what it says: Given a set X and another set f (that represents a binary function on X) we are defining f lifted to function space over X as the binary function (a set of pairs) on the space $F = X \to \text{range}(f)$ such that the value of this function on pair $\langle a, b\rangle$ of functions on X is another function c on X with values defined by $c(x) = f\langle a(x), b(x)\rangle$.

definition
Lift2FcnSpce (infix –lifted to function space over" 65) where
 f –lifted to function space over" X ≡
  –⟨ p,–⟨x,f⟨fst(p)(x),snd(p)(x)⟩⟩. x ∈ X"⟩.
  p ∈ (X→range(f))×(X→range(f))"

The result of the lift belongs to the function space.

lemma func˙ZF˙1˙L1:
  assumes A1: f : Y×Y→Y
  and A2: p ∈(X→range(f))×(X→range(f))
  shows
  –⟨x,f⟨fst(p)(x),snd(p)(x)⟩⟩. x ∈ X" : X→range(f)
  ⟨*proof*⟩

The values of the lift are defined by the value of the liftee in a natural way.

lemma func˙ZF˙1˙L2:
  assumes A1: f : Y×Y→Y
  and A2: p ∈ (X→range(f))×(X→range(f)) and A3: x∈X
  and A4: P = –⟨x,f⟨fst(p)(x),snd(p)(x)⟩⟩. x ∈ X˝
  shows P(x) = f⟨fst(p)(x),snd(p)(x)⟩
⟨proof⟩

Function lifted to a function space results in function space operator.

theorem func˙ZF˙1˙L3:
  assumes f : Y×Y→Y
  and F = f –lifted to function space over˝ X
  shows F : (X→range(f))×(X→range(f))→(X→range(f))
  ⟨proof⟩

The values of the lift are defined by the values of the liftee in the natural way.

theorem func˙ZF˙1˙L4:
  assumes A1: f : Y×Y→Y
  and A2: F = f –lifted to function space over˝ X
  and A3: s:X→range(f) r:X→range(f)
  and A4: x∈X
  shows (F⟨s,r⟩)(x) = f⟨s(x),r(x)⟩
⟨proof⟩

11.2 Associative and commutative operations

In this section we define associative and commutative operations and prove that they remain such when we lift them to a function space.

Typically we say that a binary operation "·" on a set G is "associative" if $(x \cdot y) \cdot z = x \cdot (y \cdot z)$ for all $x, y, z \in G$. Our actual definition below does not use the multiplicative notation so that we can apply it equally to the additive notation + or whatever infix symbol we may want to use. Instead, we use the generic set theory notation and write $P\langle x, y\rangle$ to denote the value of the operation P on a pair $\langle x, y\rangle \in G \times G$.

definition
  IsAssociative (infix –is associative on˝ 65) where
  P –is associative on˝ G ≡ P : G×G→G ∧
  (∀ x ∈ G. ∀ y ∈ G. ∀ z ∈ G.
  (P(⟨P(⟨x,y⟩),z⟩) = P(⟨x,P(⟨y,z⟩)⟩)))

A binary function $f : X \times X \to Y$ is commutative if $f\langle x, y\rangle = f\langle y, x\rangle$. Note that in the definition of associativity above we talk about binary "operation" and here we say use the term binary "function". This is not set in stone, but usually the word "operation" is used when the range is a factor of the domain, while the word "function" allows the range to be a completely unrelated set.

definition
  IsCommutative (infix –is commutative on″ 65) where
  f –is commutative on″ G ≡ ∀x∈G. ∀y∈G. f⟨x,y⟩ = f⟨y,x⟩

The lift of a commutative function is commutative.

lemma func˙ZF˙2˙L1:
  assumes A1: f : G×G→G
  and A2: F = f –lifted to function space over″ X
  and A3: s : X→range(f) r : X→range(f)
  and A4: f –is commutative on″ G
  shows F⟨s,r⟩ = F⟨r,s⟩
⟨*proof*⟩

The lift of a commutative function is commutative on the function space.

lemma func˙ZF˙2˙L2:
  assumes f : G×G→G
  and f –is commutative on″ G
  and F = f –lifted to function space over″ X
  shows F –is commutative on″ (X→range(f))
  ⟨*proof*⟩

The lift of an associative function is associative.

lemma func˙ZF˙2˙L3:
  assumes A2: F = f –lifted to function space over″ X
  and A3: s : X→range(f) r : X→range(f) q : X→range(f)
  and A4: f –is associative on″ G
  shows F⟨F⟨s,r⟩,q⟩ = F⟨s,F⟨r,q⟩⟩
⟨*proof*⟩

The lift of an associative function is associative on the function space.

lemma func˙ZF˙2˙L4:
  assumes A1: f –is associative on″ G
  and A2: F = f –lifted to function space over″ X
  shows F –is associative on″ (X→range(f))
⟨*proof*⟩

11.3 Restricting operations

In this section we consider conditions under which restriction of the operation to a set inherits properties like commutativity and associativity.

The commutativity is inherited when restricting a function to a set.

lemma func˙ZF˙4˙L1:
  assumes A1: f:X×X→Y and A2: A⊆X
  and A3: f –is commutative on″ X
  shows restrict(f,A×A) –is commutative on″ A
⟨*proof*⟩

Next we define what it means that a set is closed with respect to an operation.

```
definition
  IsOpClosed (infix –is closed under" 65) where
  A –is closed under" f ≡ ∀x∈A. ∀y∈A. f⟨x,y⟩ ∈ A
```

Associative operation restricted to a set that is closed with resp. to this operation is associative.

```
lemma func˙ZF˙4˙L2:assumes A1: f –is associative on" X
  and A2: A⊆X and A3: A –is closed under" f
  and A4: x∈A y∈A z∈A
  and A5: g = restrict(f,A×A)
  shows g⟨g⟨x,y⟩,z⟩ = g⟨x,g⟨y,z⟩⟩
⟨proof⟩
```

An associative operation restricted to a set that is closed with resp. to this operation is associative on the set.

```
lemma func˙ZF˙4˙L3: assumes A1: f –is associative on" X
  and A2: A⊆X and A3: A –is closed under" f
  shows restrict(f,A×A) –is associative on" A
⟨proof⟩
```

The essential condition to show that if a set A is closed with respect to an operation, then it is closed under this operation restricted to any superset of A.

```
lemma func˙ZF˙4˙L4: assumes A –is closed under" f
  and A⊆B and x∈A  y∈A and g = restrict(f,B×B)
  shows g⟨x,y⟩ ∈ A
  ⟨proof⟩
```

If a set A is closed under an operation, then it is closed under this operation restricted to any superset of A.

```
lemma func˙ZF˙4˙L5:
  assumes A1: A –is closed under" f
  and A2: A⊆B
  shows A –is closed under" restrict(f,B×B)
⟨proof⟩
```

The essential condition to show that intersection of sets that are closed with respect to an operation is closed with respect to the operation.

```
lemma func˙ZF˙4˙L6:
  assumes A –is closed under" f
  and B –is closed under" f
  and x ∈ A∩B y∈ A∩B
  shows f⟨x,y⟩ ∈ A∩B ⟨proof⟩
```

Intersection of sets that are closed with respect to an operation is closed under the operation.

lemma func˙ZF˙4˙L7:
assumes A –is closed under″ f
B –is closed under″ f
shows A∩B –is closed under″ f
⟨*proof*⟩

11.4 Compositions

For any set X we can consider a binary operation on the set of functions $f : X \to X$ defined by $C(f,g) = f \circ g$. Composition of functions (or relations) is defined in the standard Isabelle distribution as a higher order function and denoted with the letter O. In this section we consider the corresponding two-argument ZF-function (binary operation), that is a subset of $((X \to X) \times (X \to X)) \times (X \to X)$.

We define the notion of composition on the set X as the binary operation on the function space $X \to X$ that takes two functions and creates the their composition.

definition
Composition(X) ≡
–⟨p,fst(p) O snd(p)⟩. p ∈ (X→X)×(X→X)″

Composition operation is a function that maps $(X \to X) \times (X \to X)$ into $X \to X$.

lemma func˙ZF˙5˙L1: shows Composition(X) : (X→X)×(X→X)→(X→X)
⟨*proof*⟩

The value of the composition operation is the composition of arguments.

lemma func˙ZF˙5˙L2: assumes f:X→X and g:X→X
shows Composition(X)⟨f,g⟩ = f O g
⟨*proof*⟩

What is the value of a composition on an argument?

lemma func˙ZF˙5˙L3: assumes f:X→X and g:X→X and x∈X
shows (Composition(X)⟨f,g⟩)(x) = f(g(x))
⟨*proof*⟩

The essential condition to show that composition is associative.

lemma func˙ZF˙5˙L4: assumes A1: f:X→X g:X→X h:X→X
and A2: C = Composition(X)
shows C⟨C⟨f,g⟩,h⟩ = C⟨ f,C⟨g,h⟩⟩
⟨*proof*⟩

Composition is an associative operation on $X \to X$ (the space of functions that map X into itself).

lemma func˙ZF˙5˙L5: shows Composition(X) –is associative on″ (X→X)
⟨*proof*⟩

11.5 Identity function

In this section we show some additional facts about the identity function defined in the standard Isabelle's Perm theory. Note there is also image˙id˙same lemma in func1 theory.

A function that maps every point to itself is the identity on its domain.

lemma indentity˙fun: assumes A1: f:X→Y and A2:∀ x∈X. f(x)=x
 shows f = id(X)
⟨*proof*⟩

Composing a function with identity does not change the function.

lemma func˙ZF˙6˙L1A: assumes A1: f : X→X
 shows Composition(X)⟨f,id(X)⟩ = f
 Composition(X)⟨id(X),f⟩ = f
⟨*proof*⟩

An intuitively clear, but surprisingly nontrivial fact: identity is the only function from a singleton to itself.

lemma singleton˙fun˙id: shows (–x″ → –x″) = –id(–x″)″
⟨*proof*⟩

Another trivial fact: identity is the only bijection of a singleton with itself.

lemma single˙bij˙id: shows bij(–x″,–x″) = –id(–x″)″
⟨*proof*⟩

A kind of induction for the identity: if a function f is the identity on a set with a fixpoint of f removed, then it is the indentity on the whole set.

lemma id˙fixpoint˙rem: assumes A1: f:X→X and
 A2: p∈X and A3: f(p) = p and
 A4: restrict(f, X-–p″) = id(X-–p″)
 shows f = id(X)
⟨*proof*⟩

11.6 Lifting to subsets

Suppose we have a binary operation $f : X \times X \to X$ written additively as $f\langle x, y\rangle = x + y$. Such operation naturally defines another binary operation on the subsets of X that satisfies $A+B = \{x+y : x \in A, y \in B\}$. This new operation which we will call "f lifted to subsets" inherits many properties of f, such as associativity, commutativity and existence of the neutral element. This notion is useful for considering interval arithmetics.

The next definition describes the notion of a binary operation lifted to subsets. It is written in a way that might be a bit unexpected, but really it is the same as the intuitive definition, but shorter. In the definition we take a pair $p \in Pow(X) \times Pow(X)$, say $p = \langle A, B \rangle$, where $A, B \subseteq X$. Then we assign this pair of sets the set $\{f\langle x, y \rangle : x \in A, y \in B\} = \{f(x') : x' \in A \times B\}$ The set on the right hand side is the same as the image of $A \times B$ under f. In the definition we don't use A and B symbols, but write fst(p) and snd(p), resp. Recall that in Isabelle/ZF fst(p) and snd(p) denote the first and second components of an ordered pair p. See the lemma lift˙subsets˙explained for a more intuitive notation.

definition
  Lift2Subsets (infix –lifted to subsets of″ 65) where
  f –lifted to subsets of″ X ≡
  –⟨p, f(fst(p)×snd(p))⟩. p ∈ Pow(X)×Pow(X)″

The lift to subsets defines a binary operation on the subsets.

lemma lift˙subsets˙binop: assumes A1: f : X × X → Y
  shows (f –lifted to subsets of″ X) : Pow(X) × Pow(X) → Pow(Y)
⟨*proof*⟩

The definition of the lift to subsets rewritten in a more intuitive notation. We would like to write the last assertion as F⟨A,B⟩ = –f⟨x,y⟩. x ∈ A, y ∈ B″, but Isabelle/ZF does not allow such syntax.

lemma lift˙subsets˙explained: assumes A1: f : X×X → Y
  and A2: A ⊆ X B ⊆ X and A3: F = f –lifted to subsets of″ X
  shows
  F⟨A,B⟩ ⊆ Y and
  F⟨A,B⟩ = f(A×B)
  F⟨A,B⟩ = –f(p). p ∈ A×B″
  F⟨A,B⟩ = –f⟨x,y⟩ . ⟨x,y⟩ ∈ A×B″
⟨*proof*⟩

A sufficient condition for a point to belong to a result of lifting to subsets.

lemma lift˙subset˙suff: assumes A1: f : X × X → Y and
  A2: A ⊆ X B ⊆ X and A3: x∈A y∈B and
  A4: F = f –lifted to subsets of″ X
  shows f⟨x,y⟩ ∈ F⟨A,B⟩
⟨*proof*⟩

A kind of converse of lift˙subset˙apply, providing a necessary condition for a point to be in the result of lifting to subsets.

lemma lift˙subset˙nec: assumes A1: f : X × X → Y and
  A2: A ⊆ X B ⊆ X and
  A3: F = f –lifted to subsets of″ X and
  A4: z ∈ F⟨A,B⟩
  shows ∃x y. x∈A ∧ y∈B ∧ z = f⟨x,y⟩
⟨*proof*⟩

Lifting to subsets inherits commutativity.

lemma lift˙subset˙comm: assumes A1: f : X × X → Y and

A2: f –is commutative on" X and
A3: F = f –lifted to subsets of" X
shows F –is commutative on" Pow(X)
⟨*proof*⟩

Lifting to subsets inherits associativity. To show that $F\langle\langle A,B\rangle C\rangle = F\langle A, F\langle B,C\rangle\rangle$ we prove two inclusions and the proof of the second inclusion is very similar to the proof of the first one.

lemma lift˙subset˙assoc: assumes
A1: f –is associative on" X and A2: F = f –lifted to subsets of" X
shows F –is associative on" Pow(X)
⟨*proof*⟩

11.7 Distributive operations

In this section we deal with pairs of operations such that one is distributive with respect to the other, that is $a\cdot(b+c) = a\cdot b + a\cdot c$ and $(b+c)\cdot a = b\cdot a + c\cdot a$. We show that this property is preserved under restriction to a set closed with respect to both operations. In EquivClass1 theory we show that this property is preserved by projections to the quotient space if both operations are congruent with respect to the equivalence relation.

We define distributivity as a statement about three sets. The first set is the set on which the operations act. The second set is the additive operation (a ZF function) and the third is the multiplicative operation.

definition
IsDistributive(X,A,M) ≡ (∀ a∈X.∀ b∈X.∀ c∈X.
M⟨a,A⟨b,c⟩⟩ = A⟨M⟨a,b⟩,M⟨a,c⟩⟩ ∧
M⟨A⟨b,c⟩,a⟩ = A⟨M⟨b,a⟩,M⟨c,a⟩ ⟩)

The essential condition to show that distributivity is preserved by restrictions to sets that are closed with respect to both operations.

lemma func˙ZF˙7˙L1:
assumes A1: IsDistributive(X,A,M)
and A2: Y⊆X
and A3: Y –is closed under" A Y –is closed under" M
and A4: A_r = restrict(A,Y×Y) M_r = restrict(M,Y×Y)
and A5: a∈Y b∈Y c∈Y
shows M_r⟨ a,A_r⟨b,c⟩ ⟩ = A_r⟨ M_r⟨a,b⟩,M_r⟨a,c⟩ ⟩ ∧
M_r⟨ A_r⟨b,c⟩,a ⟩ = A_r⟨ M_r⟨b,a⟩, M_r⟨c,a⟩ ⟩
⟨*proof*⟩

Distributivity is preserved by restrictions to sets that are closed with respect to both operations.

lemma func˙ZF˙7˙L2:
assumes IsDistributive(X,A,M)
and Y⊆X

and Y –is closed under″ A
Y –is closed under″ M
and A_r = restrict(A,Y×Y) M_r = restrict(M,Y×Y)
shows IsDistributive(Y,A_r,M_r)
⟨*proof*⟩

end

12 More on functions

theory func˙ZF˙1 imports ZF.Order Order˙ZF˙1a func˙ZF

begin

In this theory we consider some properties of functions related to order relations

12.1 Functions and order

This section deals with functions between ordered sets.

If every value of a function on a set is bounded below by a constant, then the image of the set is bounded below.

lemma func˙ZF˙8˙L1:
assumes f:X→Y and A⊆X and ∀x∈A. ⟨L,f(x)⟩ ∈ r
shows IsBoundedBelow(f(A),r)
⟨*proof*⟩

If every value of a function on a set is bounded above by a constant, then the image of the set is bounded above.

lemma func˙ZF˙8˙L2:
assumes f:X→Y and A⊆X and ∀x∈A. ⟨f(x),U⟩ ∈ r
shows IsBoundedAbove(f(A),r)
⟨*proof*⟩

Identity is an order isomorphism.

lemma id˙ord˙iso: shows id(X) ∈ ord˙iso(X,r,X,r)
⟨*proof*⟩

Identity is the only order automorphism of a singleton.

lemma id˙ord˙auto˙singleton:
shows ord˙iso(–x″,r,–x″,r) = –id(–x″)″
⟨*proof*⟩

The image of a maximum by an order isomorphism is a maximum. Note that from the fact the r is antisymmetric and f is an order isomorphism between (A, r) and (B, R) we can not conclude that R is antisymmetric (we can only show that $R \cap (B \times B)$ is).

lemma max˙image˙ord˙iso:
  assumes A1: antisym(r) and A2: antisym(R) and
  A3: f ∈ ord˙iso(A,r,B,R) and
  A4: HasAmaximum(r,A)
  shows HasAmaximum(R,B) and Maximum(R,B) = f(Maximum(r,A))
⟨*proof*⟩

Maximum is a fixpoint of order automorphism.

lemma max˙auto˙fixpoint:
  assumes antisym(r) and f ∈ ord˙iso(A,r,A,r)
  and HasAmaximum(r,A)
  shows Maximum(r,A) = f(Maximum(r,A))
  ⟨*proof*⟩

If two sets are order isomorphic and we remove x and $f(x)$, respectively, from the sets, then they are still order isomorphic.

lemma ord˙iso˙rem˙point:
  assumes A1: f ∈ ord˙iso(A,r,B,R) and A2: a ∈ A
  shows restrict(f,A-–a") ∈ ord˙iso(A-–a",r,B-–f(a)",R)
⟨*proof*⟩

If two sets are order isomorphic and we remove maxima from the sets, then they are still order isomorphic.

corollary ord˙iso˙rem˙max:
  assumes A1: antisym(r) and f ∈ ord˙iso(A,r,B,R) and
  A4: HasAmaximum(r,A) and A5: M = Maximum(r,A)
  shows restrict(f,A-–M") ∈ ord˙iso(A-–M", r, B-–f(M)",R)
  ⟨*proof*⟩

Lemma about extending order isomorphisms by adding one point to the domain.

lemma ord˙iso˙extend: assumes A1: f ∈ ord˙iso(A,r,B,R) and
  A2: M_A ∉ A M_B ∉ B and
  A3: ∀ a∈A. ⟨a, M_A⟩ ∈ r ∀ b∈B. ⟨b, M_B⟩ ∈ R and
  A4: antisym(r) antisym(R) and
  A5: ⟨M_A,M_A⟩ ∈ r ⟷ ⟨M_B,M_B⟩ ∈ R
  shows f ∪ –⟨ M_A,M_B⟩" ∈ ord˙iso(A∪–M_A" ,r,B∪–M_B" ,R)
⟨*proof*⟩

A kind of converse to ord˙iso˙rem˙max: if two linearly ordered sets sets are order isomorphic after removing the maxima, then they are order isomorphic.

lemma rem˙max˙ord˙iso:
  assumes A1: IsLinOrder(X,r) IsLinOrder(Y,R) and
  A2: HasAmaximum(r,X) HasAmaximum(R,Y)
  ord˙iso(X - –Maximum(r,X)",r,Y - –Maximum(R,Y)",R) ≠ 0

shows ord˙iso(X,r,Y,R) $\neq$ 0
⟨*proof*⟩

12.2 Functions in cartesian products

In this section we consider maps arising naturally in cartesian products.

There is a natural bijection etween $X = Y \times \{y\}$ (a "slice") and Y. We will call this the SliceProjection(Y×–y″). This is really the ZF equivalent of the meta-function fst(x).

definition
  SliceProjection(X) ≡ –⟨p,fst(p)⟩. p ∈ X ″

A slice projection is a bijection between $X \times \{y\}$ and X.

lemma slice˙proj˙bij: shows
  SliceProjection(X×–y″): X×–y″ → X
  domain(SliceProjection(X×–y″)) = X×–y″
  ∀p∈X×–y″. SliceProjection(X×–y″)(p) = fst(p)
  SliceProjection(X×–y″) ∈ bij(X×–y″,X)
⟨*proof*⟩

Given 2 functions $f : A \to B$ and $g : C \to D$, we can consider a function $h : A \times C \to B \times D$ such that $h(x, y) = \langle f(x), g(y)\rangle$

definition
  ProdFunction where
  ProdFunction(f,g) ≡ –⟨z,⟨f(fst(z)),g(snd(z))⟩⟩. z∈domain(f)×domain(g)″

For given functions $f : A \to B$ and $g : C \to D$ the function ProdFunction(f,g) maps $A \times C$ to $B \times D$.

lemma prodFunction:
  assumes f:A→B g:C→D
  shows ProdFunction(f,g):(A×C)→(B×D)
⟨*proof*⟩

For given functions $f : A \to B$ and $g : C \to D$ and points $x \in A$, $y \in C$ the value of the function ProdFunction(f,g) on $\langle x, y\rangle$ is $\langle f(x), g(y)\rangle$.

lemma prodFunctionApp:
  assumes f:A→B g:C→D x∈A y∈C
  shows ProdFunction(f,g)⟨x,y⟩ = ⟨f(x),g(y)⟩
⟨*proof*⟩

Somewhat technical lemma about inverse image of a set by a ProdFunction(f,f).

lemma prodFunVimage: assumes x∈X f:X→Y
  shows ⟨x,t⟩ ∈ ProdFunction(f,f)-(V) ⟷ t∈X ∧ ⟨fx,ft⟩ ∈ V
⟨*proof*⟩

12.3 Induced relations and order isomorphisms

When we have two sets X, Y, function $f : X \to Y$ and a relation R on Y we can define a relation r on X by saying that $x\ r\ y$ if and only if $f(x)\ R\ f(y)$. This is especially interesting when f is a bijection as all reasonable properties of R are inherited by r. This section treats mostly the case when R is an order relation and f is a bijection. The standard Isabelle's Order theory defines the notion of a space of order isomorphisms between two sets relative to a relation. We expand that material proving that order isomrphisms preserve interesting properties of the relation.

We call the relation created by a relation on Y and a mapping $f : X \to Y$ the InducedRelation(f,R).

definition
  InducedRelation(f,R) ≡
  –p ∈ domain(f)×domain(f). ⟨f(fst(p)),f(snd(p))⟩ ∈ R″

A reformulation of the definition of the relation induced by a function.

lemma def˙of˙ind˙relA:
  assumes ⟨x,y⟩ ∈ InducedRelation(f,R)
  shows ⟨f(x),f(y)⟩ ∈ R
  ⟨*proof*⟩

A reformulation of the definition of the relation induced by a function, kind of converse of def˙of˙ind˙relA.

lemma def˙of˙ind˙relB: assumes f:A→B and
  x∈A y∈A and ⟨f(x),f(y)⟩ ∈ R
  shows ⟨x,y⟩ ∈ InducedRelation(f,R)
  ⟨*proof*⟩

A property of order isomorphisms that is missing from standard Isabelle's Order.thy.

lemma ord˙iso˙apply˙conv:
  assumes f ∈ ord˙iso(A,r,B,R) and
  ⟨f(x),f(y)⟩ ∈ R and x∈A y∈A
  shows ⟨x,y⟩ ∈ r
  ⟨*proof*⟩

The next lemma tells us where the induced relation is defined

lemma ind˙rel˙domain:
  assumes R ⊆ B×B and f:A→B
  shows InducedRelation(f,R) ⊆ A×A
  ⟨*proof*⟩

A bijection is an order homomorphisms between a relation and the induced one.

lemma bij˙is˙ord˙iso: assumes A1: f ∈ bij(A,B)

shows f ∈ ord_iso(A,InducedRelation(f,R),B,R)
⟨*proof*⟩

An order isomoprhism preserves antisymmetry.

lemma ord_iso_pres_antsym: assumes A1: f ∈ ord_iso(A,r,B,R) and
A2: r ⊆ A×A and A3: antisym(R)
shows antisym(r)
⟨*proof*⟩

Order isomoprhisms preserve transitivity.

lemma ord_iso_pres_trans: assumes A1: f ∈ ord_iso(A,r,B,R) and
A2: r ⊆ A×A and A3: trans(R)
shows trans(r)
⟨*proof*⟩

Order isomorphisms preserve totality.

lemma ord_iso_pres_tot: assumes A1: f ∈ ord_iso(A,r,B,R) and
A2: r ⊆ A×A and A3: R {is total on} B
shows r {is total on} A
⟨*proof*⟩

Order isomorphisms preserve linearity.

lemma ord_iso_pres_lin: assumes f ∈ ord_iso(A,r,B,R) and
r ⊆ A×A and IsLinOrder(B,R)
shows IsLinOrder(A,r)
⟨*proof*⟩

If a relation is a linear order, then the relation induced on another set by a bijection is also a linear order.

lemma ind_rel_pres_lin:
assumes A1: f ∈ bij(A,B) and A2: IsLinOrder(B,R)
shows IsLinOrder(A,InducedRelation(f,R))
⟨*proof*⟩

The image by an order isomorphism of a bounded above and nonempty set is bounded above.

lemma ord_iso_pres_bound_above:
assumes A1: f ∈ ord_iso(A,r,B,R) and A2: r ⊆ A×A and
A3: IsBoundedAbove(C,r) C≠0
shows IsBoundedAbove(f(C),R) f(C) ≠ 0
⟨*proof*⟩

Order isomorphisms preserve the property of having a minimum.

lemma ord_iso_pres_has_min:
assumes A1: f ∈ ord_iso(A,r,B,R) and A2: r ⊆ A×A and
A3: C⊆A and A4: HasAminimum(R,f(C))
shows HasAminimum(r,C)

⟨*proof*⟩

Order isomorhisms preserve the images of relations. In other words taking the image of a point by a relation commutes with the function.

lemma ord˙iso˙pres˙rel˙image:
  assumes A1: f ∈ ord˙iso(A,r,B,R) and
  A2: r ⊆ A×A R ⊆ B×B and
  A3: a∈A
  shows f(r–a″) = R–f(a)″
⟨*proof*⟩

Order isomorphisms preserve collections of upper bounds.

lemma ord˙iso˙pres˙up˙bounds:
  assumes A1: f ∈ ord˙iso(A,r,B,R) and
  A2: r ⊆ A×A R ⊆ B×B and
  A3: C⊆A
  shows –f(r–a″). a∈C″ = –R–b″. b ∈ f(C)″
⟨*proof*⟩

The image of the set of upper bounds is the set of upper bounds of the image.

lemma ord˙iso˙pres˙min˙up˙bounds:
  assumes A1: f ∈ ord˙iso(A,r,B,R) and A2: r ⊆ A×A R ⊆ B×B and
  A3: C⊆A and A4: C≠0
  shows f(⋂a∈C. r–a″) = (⋂b∈f(C). R–b″)
⟨*proof*⟩

Order isomorphisms preserve completeness.

lemma ord˙iso˙pres˙compl:
  assumes A1: f ∈ ord˙iso(A,r,B,R) and
  A2: r ⊆ A×A R ⊆ B×B and A3: R –is complete″
  shows r –is complete″
⟨*proof*⟩

If the original relation is complete, then the induced one is complete.

lemma ind˙rel˙pres˙compl: assumes A1: f ∈ bij(A,B)
  and A2: R ⊆ B×B and A3: R –is complete″
  shows InducedRelation(f,R) –is complete″
⟨*proof*⟩

end

13 Finite sets - introduction

theory Finite˙ZF imports ZF1 Nat˙ZF˙IML ZF.Cardinal

begin

Standard Isabelle Finite.thy contains a very useful notion of finite powerset: the set of finite subsets of a given set. The definition, however, is specific to Isabelle and based on the notion of "datatype", obviously not something that belongs to ZF set theory. This theory file devolops the notion of finite powerset similarly as in Finite.thy, but based on standard library's Cardinal.thy. This theory file is intended to replace IsarMathLib's Finite1 and Finite˙ZF˙1 theories that are currently derived from the "datatype" approach.

13.1 Definition and basic properties of finite powerset

The goal of this section is to prove an induction theorem about finite powersets: if the empty set has some property and this property is preserved by adding a single element of a set, then this property is true for all finite subsets of this set.

We defined the finite powerset FinPow(X) as those elements of the powerset that are finite.

definition
  FinPow(X) ≡ –A ∈ Pow(X). Finite(A)″

The cardinality of an element of finite powerset is a natural number.

lemma card˙fin˙is˙nat: assumes A ∈ FinPow(X)
  shows —A— ∈ nat and A ≈ —A—
  ⟨*proof*⟩

A reformulation of card˙fin˙is˙nat: for a finit set A there is a bijection between $|A|$ and A.

lemma fin˙bij˙card: assumes A1: A ∈ FinPow(X)
  shows ∃ b. b ∈ bij(—A—, A)
⟨*proof*⟩

If a set has the same number of elements as $n \in \mathbb{N}$, then its cardinality is n. Recall that in set theory a natural number n is a set that has n elements.

lemma card˙card: assumes A ≈ n and n ∈ nat
  shows —A— = n
  ⟨*proof*⟩

If we add a point to a finite set, the cardinality increases by one. To understand the second assertion $|A \cup \{a\}| = |A| \cup \{|A|\}$ recall that the cardinality $|A|$ of A is a natural number and for natural numbers we have $n+1 = n \cup \{n\}$.

lemma card˙fin˙add˙one: assumes A1: A ∈ FinPow(X) and A2: a ∈ X-A
  shows
  —A ∪ –a″— = succ(—A—)
  —A ∪ –a″— = —A— ∪ ——A—″

⟨*proof*⟩

We can decompose the finite powerset into collection of sets of the same natural cardinalities.

lemma finpow_decomp:
  shows FinPow(X) = $(\bigcup n \in nat. \{A \in Pow(X).\ A \approx n\})$
  ⟨*proof*⟩

Finite powerset is the union of sets of cardinality bounded by natural numbers.

lemma finpow_union_card_nat:
  shows FinPow(X) = $(\bigcup n \in nat. \{A \in Pow(X).\ A \lesssim n\})$
⟨*proof*⟩

A different form of finpow_union_card_nat (see above) - a subset that has not more elements than a given natural number is in the finite powerset.

lemma lepoll_nat_in_finpow:
  assumes $n \in nat$ $A \subseteq X$ $A \lesssim n$
  shows $A \in$ FinPow(X)
  ⟨*proof*⟩

Natural numbers are finite subsets of the set of natural numbers.

lemma nat_finpow_nat: assumes $n \in nat$ shows $n \in$ FinPow(nat)
  ⟨*proof*⟩

A finite subset is a finite subset of itself.

lemma fin_finpow_self: assumes $A \in$ FinPow(X) shows $A \in$ FinPow(A)
  ⟨*proof*⟩

If we remove an element and put it back we get the set back.

lemma rem_add_eq: assumes $a \in A$ shows $(A-\{a\}) \cup \{a\} = A$
  ⟨*proof*⟩

Induction for finite powerset. This is smilar to the standard Isabelle's Fin_induct.

theorem FinPow_induct: assumes A1: P(0) and
  A2: $\forall A \in$ FinPow(X). P(A) $\longrightarrow$ ($\forall a \in X$. P($A \cup \{a\}$)) and
  A3: $B \in$ FinPow(X)
  shows P(B)
⟨*proof*⟩

A subset of a finite subset is a finite subset.

lemma subset_finpow: assumes $A \in$ FinPow(X) and $B \subseteq A$
  shows $B \in$ FinPow(X)
  ⟨*proof*⟩

If we subtract anything from a finite set, the resulting set is finite.

lemma diff˙finpow:
assumes A ∈ FinPow(X) shows A-B ∈ FinPow(X)
⟨*proof*⟩

If we remove a point from a finites subset, we get a finite subset.

corollary fin˙rem˙point˙fin: assumes A ∈ FinPow(X)
shows A - –a″ ∈ FinPow(X)
⟨*proof*⟩

Cardinality of a nonempty finite set is a successsor of some natural number.

lemma card˙non˙empty˙succ:
assumes A1: A ∈ FinPow(X) and A2: A ≠ 0
shows ∃ n ∈ nat. —A— = succ(n)
⟨*proof*⟩

Nonempty set has non-zero cardinality. This is probably true without the assumption that the set is finite, but I couldn't derive it from standard Isabelle theorems.

lemma card˙non˙empty˙non˙zero:
assumes A ∈ FinPow(X) and A ≠ 0
shows —A— ≠ 0
⟨*proof*⟩

Another variation on the induction theme: If we can show something holds for the empty set and if it holds for all finite sets with at most k elements then it holds for all finite sets with at most $k + 1$ elements, the it holds for all finite sets.

theorem FinPow˙card˙ind: assumes A1: P(0) and
A2: ∀ k∈nat.
(∀ A ∈ FinPow(X). A $\lesssim$ k ⟶ P(A)) ⟶
(∀ A ∈ FinPow(X). A $\lesssim$ succ(k) ⟶ P(A))
and A3: A ∈ FinPow(X) shows P(A)
⟨*proof*⟩

Another type of induction (or, maybe recursion). In the induction step we try to find a point in the set that if we remove it, the fact that the property holds for the smaller set implies that the property holds for the whole set.

lemma FinPow˙ind˙rem˙one: assumes A1: P(0) and
A2: ∀ A ∈ FinPow(X). A ≠ 0 ⟶ (∃ a∈A. P(A-–a″) ⟶ P(A))
and A3: B ∈ FinPow(X)
shows P(B)
⟨*proof*⟩

Yet another induction theorem. This is similar, but slightly more complicated than FinPow˙ind˙rem˙one. The difference is in the treatment of the empty set to allow to show properties that are not true for empty set.

lemma FinPow˙rem˙ind: assumes A1: ∀ A ∈ FinPow(X).

A = 0 ∨ (∃ a∈A. A = –a″ ∨ P(A-–a″) ⟶ P(A))
and A2: A ∈ FinPow(X) and A3: A≠0
shows P(A)
⟨*proof*⟩

If a family of sets is closed with respect to taking intersections of two sets then it is closed with respect to taking intersections of any nonempty finite collection.

lemma inter˙two˙inter˙fin:
assumes A1: ∀ V∈T. ∀ W∈T. V ∩ W ∈ T and
A2: N ≠ 0 and A3: N ∈ FinPow(T)
shows (⋂ N ∈ T)
⟨*proof*⟩

If a family of sets contains the empty set and is closed with respect to taking unions of two sets then it is closed with respect to taking unions of any finite collection.

lemma union˙two˙union˙fin:
assumes A1: 0 ∈ C and A2: ∀ A∈C. ∀ B∈C. A∪B ∈ C and
A3: N ∈ FinPow(C)
shows ⋃ N ∈ C
⟨*proof*⟩

Empty set is in finite power set.

lemma empty˙in˙finpow: shows 0 ∈ FinPow(X)
⟨*proof*⟩

Singleton is in the finite powerset.

lemma singleton˙in˙finpow: assumes x ∈ X
shows –x″ ∈ FinPow(X) ⟨*proof*⟩

Union of two finite subsets is a finite subset.

lemma union˙finpow: assumes A ∈ FinPow(X) and B ∈ FinPow(X)
shows A ∪ B ∈ FinPow(X)
⟨*proof*⟩

Union of finite number of finite sets is finite.

lemma fin˙union˙finpow: assumes M ∈ FinPow(FinPow(X))
shows ⋃ M ∈ FinPow(X)
⟨*proof*⟩

If a set is finite after removing one element, then it is finite.

lemma rem˙point˙fin˙fin:
assumes A1: x ∈ X and A2: A - –x″ ∈ FinPow(X)
shows A ∈ FinPow(X)
⟨*proof*⟩

An image of a finite set is finite.

lemma fin˙image˙fin: assumes ∀ V∈B. K(V)∈C and N ∈ FinPow(B)
shows –K(V). V∈N″ ∈ FinPow(C)
⟨*proof*⟩

Union of a finite indexed family of finite sets is finite.

lemma union˙fin˙list˙fin:
assumes A1: n ∈ nat and A2: ∀ k ∈ n. N(k) ∈ FinPow(X)
shows
–N(k). k ∈ n″ ∈ FinPow(FinPow(X)) and (⋃k ∈ n. N(k)) ∈ FinPow(X)
⟨*proof*⟩

end

14 Finite sets

theory Finite1 imports ZF.EquivClass ZF.Finite func1 ZF1

begin

This theory extends Isabelle standard Finite theory. It is obsolete and should not be used for new development. Use the Finite˙ZF instead.

14.1 Finite powerset

In this section we consider various properties of Fin datatype (even though there are no datatypes in ZF set theory).

In Topology˙ZF theory we consider induced topology that is obtained by taking a subset of a topological space. To show that a topology restricted to a subset is also a topology on that subset we may need a fact that if T is a collection of sets and A is a set then every finite collection $\{V_i\}$ is of the form $V_i = U_i \cap A$, where $\{U_i\}$ is a finite subcollection of T. This is one of those trivial facts that require suprisingly long formal proof. Actually, the need for this fact is avoided by requiring intersection two open sets to be open (rather than intersection of a finite number of open sets). Still, the fact is left here as an example of a proof by induction. We will use Fin˙induct lemma from Finite.thy. First we define a property of finite sets that we want to show.

definition
Prfin(T,A,M) ≡ ((M = 0) — (∃N∈ Fin(T). ∀V∈ M. ∃ U∈ N. (V = U∩A)))

Now we show the main induction step in a separate lemma. This will make the proof of the theorem FinRestr below look short and nice. The premises of the ind˙step lemma are those needed by the main induction step in lemma Fin˙induct (see standard Isabelle's Finite.thy).

lemma ind˙step: assumes A: ∀ V∈ TA. ∃ U∈T. V=U∩A

and A1: $W\in TA$ and A2: $M\in$ Fin(TA)
and A3: $W\notin M$ and A4: Prfin(T,A,M)
shows Prfin(T,A,cons(W,M))
⟨*proof*⟩

Now we are ready to prove the statement we need.

theorem FinRestr0: assumes A: $\forall\ V \in TA.\ \exists\ U\in T.\ V=U\cap A$
shows $\forall\ M\in$ Fin(TA). Prfin(T,A,M)
⟨*proof*⟩

This is a different form of the above theorem:

theorem ZF1FinRestr:
assumes A1:$M\in$ Fin(TA) and A2: $M\neq 0$
and A3: $\forall\ V\in TA.\ \exists\ U\in T.\ V=U\cap A$
shows $\exists N\in$ Fin(T). $(\forall V\in M.\ \exists\ U\in N.\ (V = U\cap A)) \wedge N\neq 0$
⟨*proof*⟩

Purely technical lemma used in Topology_ZF_1 to show that if a topology is T_2, then it is T_1.

lemma Finite1_L2:
assumes A:$\exists U\ V.\ (U\in T \wedge V\in T \wedge x\in U \wedge y\in V \wedge U\cap V=0)$
shows $\exists U\in T.\ (x\in U \wedge y\notin U)$
⟨*proof*⟩

A collection closed with respect to taking a union of two sets is closed under taking finite unions. Proof by induction with the induction step formulated in a separate lemma.

lemma Finite1_L3_IndStep:
assumes A1:$\forall A\ B.\ ((A\in C \wedge B\in C) \longrightarrow A\cup B\in C)$
and A2: $A\in C$ and A3: $N\in$Fin(C) and A4:$A\notin N$ and A5:$\bigcup N \in C$
shows $\bigcup$cons(A,N) $\in C$
⟨*proof*⟩

The lemma: a collection closed with respect to taking a union of two sets is closed under taking finite unions.

lemma Finite1_L3:
assumes A1: $0 \in C$ and A2: $\forall A\ B.\ ((A\in C \wedge B\in C) \longrightarrow A\cup B\in C)$ and
A3: $N\in$ Fin(C)
shows $\bigcup N\in C$
⟨*proof*⟩

A collection closed with respect to taking a intersection of two sets is closed under taking finite intersections. Proof by induction with the induction step formulated in a separate lemma. This is sligltly more involved than the union case in Finite1_L3, because the intersection of empty collection is undefined (or should be treated as such). To simplify notation we define the property to be proven for finite sets as a separate notion.

definition
  $\text{IntPr}(T,N) \equiv (N = 0 — \bigcap N \in T)$

The induction step.

lemma Finite1˙L4˙IndStep:
  assumes A1: $\forall A\ B.\ ((A\in T \wedge B\in T) \longrightarrow A\cap B\in T)$
  and A2: $A\in T$ and A3:$N\in \text{Fin}(T)$ and A4:$A\notin N$ and A5:$\text{IntPr}(T,N)$
  shows $\text{IntPr}(T,\text{cons}(A,N))$
⟨*proof*⟩

The lemma.

lemma Finite1˙L4:
  assumes A1: $\forall A\ B.\ A\in T \wedge B\in T \longrightarrow A\cap B \in T$
  and A2: $N\in \text{Fin}(T)$
  shows $\text{IntPr}(T,N)$
⟨*proof*⟩

Next is a restatement of the above lemma that does not depend on the IntPr meta-function.

lemma Finite1˙L5:
  assumes A1: $\forall A\ B.\ ((A\in T \wedge B\in T) \longrightarrow A\cap B\in T)$
  and A2: $N\neq 0$ and A3: $N\in \text{Fin}(T)$
  shows $\bigcap N \in T$
⟨*proof*⟩

The images of finite subsets by a meta-function are finite. For example in topology if we have a finite collection of sets, then closing each of them results in a finite collection of closed sets. This is a very useful lemma with many unexpected applications. The proof is by induction. The next lemma is the induction step.

lemma fin˙image˙fin˙IndStep:
  assumes $\forall V\in B.\ K(V)\in C$
  and $U\in B$ and $N\in \text{Fin}(B)$ and $U\notin N$ and –K(V). $V\in N''\in \text{Fin}(C)$
  shows –K(V). $V\in \text{cons}(U,N)'' \in \text{Fin}(C)$
  ⟨*proof*⟩

The lemma:

lemma fin˙image˙fin:
  assumes A1: $\forall V\in B.\ K(V)\in C$ and A2: $N\in \text{Fin}(B)$
  shows –K(V). $V\in N'' \in \text{Fin}(C)$
⟨*proof*⟩

The image of a finite set is finite.

lemma Finite1˙L6A: assumes A1: $f:X\rightarrow Y$ and A2: $N \in \text{Fin}(X)$
  shows $f(N) \in \text{Fin}(Y)$
⟨*proof*⟩

If the set defined by a meta-function is finite, then every set defined by a composition of this meta function with another one is finite.

lemma Finite1˙L6B:
  assumes A1: ∀x∈X. a(x) ∈ Y and A2: –b(y).y∈Y″ ∈ Fin(Z)
  shows –b(a(x)).x∈X″ ∈ Fin(Z)
⟨*proof*⟩

If the set defined by a meta-function is finite, then every set defined by a composition of this meta function with another one is finite.

lemma Finite1˙L6C:
  assumes A1: ∀y∈Y. b(y) ∈ Z and A2: –a(x). x∈X″ ∈ Fin(Y)
  shows –b(a(x)).x∈X″ ∈ Fin(Z)
⟨*proof*⟩

Cartesian product of finite sets is finite.

lemma Finite1˙L12: assumes A1: A ∈ Fin(A) and A2: B ∈ Fin(B)
  shows A×B ∈ Fin(A×B)
⟨*proof*⟩

We define the characterisic meta-function that is the identity on a set and assigns a default value everywhere else.

definition
  Characteristic(A,default,x) ≡ (if x∈A then x else default)

A finite subset is a finite subset of itself.

lemma Finite1˙L13:
  assumes A1:A ∈ Fin(X) shows A ∈ Fin(A)
⟨*proof*⟩

Cartesian product of finite subsets is a finite subset of cartesian product.

lemma Finite1˙L14: assumes A1: A ∈ Fin(X) B ∈ Fin(Y)
  shows A×B ∈ Fin(X×Y)
⟨*proof*⟩

The next lemma is needed in the Group˙ZF˙3 theory in a couple of places.

lemma Finite1˙L15:
  assumes A1: –b(x). x∈A″ ∈ Fin(B) –c(x). x∈A″ ∈ Fin(C)
  and A2: f : B×C→E
  shows –f⟨ b(x),c(x)⟩. x∈A″ ∈ Fin(E)
⟨*proof*⟩

Singletons are in the finite powerset.

lemma Finite1˙L16: assumes x∈X shows –x″ ∈ Fin(X)
  ⟨*proof*⟩

A special case of Finite1˙L15 where the second set is a singleton. In Group˙ZF˙3 theory this corresponds to the situation where we multiply by a constant.

lemma Finite1˙L16AA: assumes –b(x). x∈A″ ∈ Fin(B)
  and c∈C and f : B×C→E
  shows –f⟨ b(x),c⟩. x∈A″ ∈ Fin(E)
⟨*proof*⟩

First order version of the induction for the finite powerset.

lemma Finite1˙L16B: assumes A1: P(0) and A2: B∈Fin(X)
  and A3: ∀A∈Fin(X).∀x∈X. x∉A ∧ P(A)⟶P(A∪–x″)
  shows P(B)
⟨*proof*⟩

14.2 Finite range functions

In this section we define functions $f : X \to Y$, with the property that $f(X)$ is a finite subset of Y. Such functions play a important role in the construction of real numbers in the Real˙ZF series.

Definition of finite range functions.

definition
  FinRangeFunctions(X,Y) ≡ –f:X→Y. f(X) ∈ Fin(Y)″

Constant functions have finite range.

lemma Finite1˙L17: assumes c∈Y and X≠0
  shows ConstantFunction(X,c) ∈ FinRangeFunctions(X,Y)
  ⟨*proof*⟩

Finite range functions have finite range.

lemma Finite1˙L18: assumes f ∈ FinRangeFunctions(X,Y)
  shows –f(x). x∈X″ ∈ Fin(Y)
  ⟨*proof*⟩

An alternative form of the definition of finite range functions.

lemma Finite1˙L19: assumes f:X→Y
  and –f(x). x∈X″ ∈ Fin(Y)
  shows f ∈ FinRangeFunctions(X,Y)
  ⟨*proof*⟩

A composition of a finite range function with another function is a finite range function.

lemma Finite1˙L20: assumes A1:f ∈ FinRangeFunctions(X,Y)
  and A2: g : Y→Z
  shows g O f ∈ FinRangeFunctions(X,Z)
⟨*proof*⟩

Image of any subset of the domain of a finite range function is finite.

lemma Finite1˙L21:
  assumes f ∈ FinRangeFunctions(X,Y) and A⊆X

shows f(A) ∈ Fin(Y)
⟨*proof*⟩

end

15 Finite sets 1

theory Finite˙ZF˙1 imports Finite1 Order˙ZF˙1a

begin

This theory is based on Finite1 theory and is obsolete. It contains properties of finite sets related to order relations. See the FinOrd theory for a better approach.

15.1 Finite vs. bounded sets

The goal of this section is to show that finite sets are bounded and have maxima and minima.

Finite set has a maximum - induction step.

lemma Finite˙ZF˙1˙L1:
assumes A1: r –is total on" X and A2: trans(r)
and A3: A∈Fin(X) and A4: x∈X and A5: A=0 ∨ HasAmaximum(r,A)
shows A∪–x" = 0 ∨ HasAmaximum(r,A∪–x")
⟨*proof*⟩

For total and transitive relations finite set has a maximum.

theorem Finite˙ZF˙1˙T1A:
assumes A1: r –is total on" X and A2: trans(r)
and A3: B∈Fin(X)
shows B=0 ∨ HasAmaximum(r,B)
⟨*proof*⟩

Finite set has a minimum - induction step.

lemma Finite˙ZF˙1˙L2:
assumes A1: r –is total on" X and A2: trans(r)
and A3: A∈Fin(X) and A4: x∈X and A5: A=0 ∨ HasAminimum(r,A)
shows A∪–x" = 0 ∨ HasAminimum(r,A∪–x")
⟨*proof*⟩

For total and transitive relations finite set has a minimum.

theorem Finite˙ZF˙1˙T1B:
assumes A1: r –is total on" X and A2: trans(r)
and A3: B ∈ Fin(X)
shows B=0 ∨ HasAminimum(r,B)
⟨*proof*⟩

For transitive and total relations finite sets are bounded.

theorem Finite˙ZF˙1˙T1:
  assumes A1: r –is total on˝ X and A2: trans(r)
  and A3: B∈Fin(X)
  shows IsBounded(B,r)
⟨*proof*⟩

For linearly ordered finite sets maximum and minimum have desired properties. The reason we need linear order is that we need the order to be total and transitive for the finite sets to have a maximum and minimum and then we also need antisymmetry for the maximum and minimum to be unique.

theorem Finite˙ZF˙1˙T2:
  assumes A1: IsLinOrder(X,r) and A2: A ∈ Fin(X) and A3: A≠0
  shows
  Maximum(r,A) ∈ A
  Minimum(r,A) ∈ A
  ∀x∈A. ⟨x,Maximum(r,A)⟩ ∈ r
  ∀x∈A. ⟨Minimum(r,A),x⟩ ∈ r
⟨*proof*⟩

A special case of Finite˙ZF˙1˙T2 when the set has three elements.

corollary Finite˙ZF˙1˙L2A:
  assumes A1: IsLinOrder(X,r) and A2: a∈X b∈X c∈X
  shows
  Maximum(r,–a,b,c˝) ∈ –a,b,c˝
  Minimum(r,–a,b,c˝) ∈ –a,b,c˝
  Maximum(r,–a,b,c˝) ∈ X
  Minimum(r,–a,b,c˝) ∈ X
  ⟨a,Maximum(r,–a,b,c˝)⟩ ∈ r
  ⟨b,Maximum(r,–a,b,c˝)⟩ ∈ r
  ⟨c,Maximum(r,–a,b,c˝)⟩ ∈ r
⟨*proof*⟩

If for every element of X we can find one in A that is greater, then the A can not be finite. Works for relations that are total, transitive and antisymmetric.

lemma Finite˙ZF˙1˙1˙L3:
  assumes A1: r –is total on˝ X
  and A2: trans(r) and A3: antisym(r)
  and A4: r ⊆ X×X and A5: X≠0
  and A6: ∀x∈X. ∃a∈A. x≠a ∧ ⟨x,a⟩ ∈ r
  shows A ∉ Fin(X)
⟨*proof*⟩

end

16 Finite sets and order relations

theory FinOrd˙ZF imports Finite˙ZF func˙ZF˙1

begin

This theory file contains properties of finite sets related to order relations. Part of this is similar to what is done in Finite˙ZF˙1 except that the development is based on the notion of finite powerset defined in Finite˙ZF rather the one defined in standard Isabelle Finite theory.

16.1 Finite vs. bounded sets

The goal of this section is to show that finite sets are bounded and have maxima and minima.

For total and transitive relations nonempty finite set has a maximum.

theorem fin˙has˙max:
  assumes A1: r –is total on″ X and A2: trans(r)
  and A3: B ∈ FinPow(X) and A4: B ≠ 0
  shows HasAmaximum(r,B)
⟨*proof*⟩

For linearly ordered nonempty finite sets the maximum is in the set and indeed it is the greatest element of the set.

lemma linord˙max˙props: assumes A1: IsLinOrder(X,r) and
  A2: A ∈ FinPow(X) A ≠ 0
  shows
  Maximum(r,A) ∈ A
  Maximum(r,A) ∈ X
  ∀a∈A. ⟨a,Maximum(r,A)⟩ ∈ r
⟨*proof*⟩

16.2 Order isomorphisms of finite sets

In this section we eastablish that if two linearly ordered finite sets have the same number of elements, then they are order-isomorphic and the isomorphism is unique. This allows us to talk about "enumeration" of a linearly ordered finite set. We define the enumeration as the order isomorphism between the number of elements of the set (which is a natural number $n = \{0, 1, .., n - 1\}$) and the set.

A really weird corner case - empty set is order isomorphic with itself.

lemma empty˙ord˙iso: shows ord˙iso(0,r,0,R) ≠ 0
⟨*proof*⟩

Even weirder than empty˙ord˙iso The order automorphism of the empty set is unique.

lemma empty˙ord˙iso˙uniq:
  assumes $f \in$ ord˙iso(0,r,0,R) $g \in$ ord˙iso(0,r,0,R)
  shows $f = g$
⟨*proof*⟩

The empty set is the only order automorphism of itself.

lemma empty˙ord˙iso˙empty: shows ord˙iso(0,r,0,R) = –0″
⟨*proof*⟩

An induction (or maybe recursion?) scheme for linearly ordered sets. The induction step is that we show that if the property holds when the set is a singleton or for a set with the maximum removed, then it holds for the set. The idea is that since we can build any finite set by adding elements on the right, then if the property holds for the empty set and is invariant with respect to this operation, then it must hold for all finite sets.

lemma fin˙ord˙induction:
  assumes A1: IsLinOrder(X,r) and A2: P(0) and
  A3: $\forall$ A $\in$ FinPow(X). A $\neq$ 0 $\longrightarrow$ (P(A - –Maximum(r,A)″) $\longrightarrow$ P(A))
  and A4: B $\in$ FinPow(X) shows P(B)
⟨*proof*⟩

A sligltly more complicated version of fin˙ord˙induction that allows to prove properties that are not true for the empty set.

lemma fin˙ord˙ind:
  assumes A1: IsLinOrder(X,r) and A2: $\forall$ A $\in$ FinPow(X).
  A = 0 $\vee$ (A = –Maximum(r,A)″ $\vee$ P(A - –Maximum(r,A)″) $\longrightarrow$ P(A))
  and A3: B $\in$ FinPow(X) and A4: B$\neq$0
  shows P(B)
⟨*proof*⟩

Yet another induction scheme. We build a linearly ordered set by adding elements that are greater than all elements in the set.

lemma fin˙ind˙add˙max:
  assumes A1: IsLinOrder(X,r) and A2: P(0) and A3: $\forall$ A $\in$ FinPow(X).
  ($\forall$ x $\in$ X-A. P(A) $\wedge$ ($\forall$a$\in$A. ⟨a,x⟩ $\in$ r) $\longrightarrow$ P(A $\cup$ –x″))
  and A4: B $\in$ FinPow(X)
  shows P(B)
⟨*proof*⟩

The only order automorphism of a linearly ordered finite set is the identity.

theorem fin˙ord˙auto˙id: assumes A1: IsLinOrder(X,r)
  and A2: B $\in$ FinPow(X) and A3: B$\neq$0
  shows ord˙iso(B,r,B,r) = –id(B)″
⟨*proof*⟩

Every two finite linearly ordered sets are order isomorphic. The statement is formulated to make the proof by induction on the size of the set easier, see fin˙ord˙iso˙ex for an alternative formulation.

lemma fin˙order˙iso:
  assumes A1: IsLinOrder(X,r) IsLinOrder(Y,R) and
  A2: n ∈ nat
  shows ∀ A ∈ FinPow(X). ∀ B ∈ FinPow(Y).
  A ≈ n ∧ B ≈ n ⟶ ord˙iso(A,r,B,R) ≠ 0
⟨*proof*⟩

Every two finite linearly ordered sets are order isomorphic.

lemma fin˙ord˙iso˙ex:
  assumes A1: IsLinOrder(X,r) IsLinOrder(Y,R) and
  A2: A ∈ FinPow(X) B ∈ FinPow(Y) and A3: B ≈ A
  shows ord˙iso(A,r,B,R) ≠ 0
⟨*proof*⟩

Existence and uniqueness of order isomorphism for two linearly ordered sets with the same number of elements.

theorem fin˙ord˙iso˙ex˙uniq:
  assumes A1: IsLinOrder(X,r) IsLinOrder(Y,R) and
  A2: A ∈ FinPow(X) B ∈ FinPow(Y) and A3: B ≈ A
  shows ∃!f. f ∈ ord˙iso(A,r,B,R)
⟨*proof*⟩

end

17 Equivalence relations

theory EquivClass1 imports ZF.EquivClass func˙ZF ZF1

begin

In this theory file we extend the work on equivalence relations done in the standard Isabelle's EquivClass theory. That development is very good and all, but we really would prefer an approach contained within the a standard ZF set theory, without extensions specific to Isabelle. That is why this theory is written.

17.1 Congruent functions and projections on the quotient

Suppose we have a set X with a relation $r \subseteq X \times X$ and a function $f : X \to X$. The function f can be compatible (congruent) with r in the sense that if two elements x, y are related then the values $f(x), f(x)$ are also related. This is especially useful if r is an equivalence relation as it allows to "project" the function to the quotient space X/r (the set of equivalence classes of r) and create a new function F that satifies the formula $F([x]_r) = [f(x)]_r$.

When f is congruent with respect to r such definition of the value of F on the equivalence class $[x]_r$ does not depend on which x we choose to represent the class. In this section we also consider binary operations that are congruent with respect to a relation. These are important in algebra - the congruency condition allows to project the operation to obtain the operation on the quotient space.

First we define the notion of function that maps equivalent elements to equivalent values. We use similar names as in the Isabelle's standard EquivClass theory to indicate the conceptual correspondence of the notions.

definition
  Congruent(r,f) ≡
  (∀x y. ⟨x,y⟩ ∈ r ⟶ ⟨f(x),f(y)⟩ ∈ r)

Now we will define the projection of a function onto the quotient space. In standard math the equivalence class of x with respect to relation r is usually denoted $[x]_r$. Here we reuse notation $r\{x\}$ instead. This means the image of the set $\{x\}$ with respect to the relation, which, for equivalence relations is exactly its equivalence class if you think about it.

definition
  ProjFun(A,r,f) ≡
  –⟨c,⋃x∈c. r–f(x)″⟩. c ∈ (A//r)″

Elements of equivalence classes belong to the set.

lemma EquivClass˙1˙L1:
  assumes A1: equiv(A,r) and A2: C ∈ A//r and A3: x∈C
  shows x∈A
⟨*proof*⟩

The image of a subset of X under projection is a subset of A/r.

lemma EquivClass˙1˙L1A:
  assumes A⊆X shows –r–x″. x∈A″ ⊆ X//r
  ⟨*proof*⟩

If an element belongs to an equivalence class, then its image under relation is this equivalence class.

lemma EquivClass˙1˙L2:
  assumes A1: equiv(A,r) C ∈ A//r and A2: x∈C
  shows r–x″ = C
⟨*proof*⟩

Elements that belong to the same equivalence class are equivalent.

lemma EquivClass˙1˙L2A:
  assumes equiv(A,r) C ∈ A//r x∈C y∈C
  shows ⟨x,y⟩ ∈ r
  ⟨*proof*⟩

Every x is in the class of y, then they are equivalent.

lemma EquivClass˙1˙L2B:
  assumes A1: equiv(A,r) and A2: y∈A and A3: x ∈ r–y″
  shows ⟨x,y⟩ ∈ r
⟨*proof*⟩

If a function is congruent then the equivalence classes of the values that come from the arguments from the same class are the same.

lemma EquivClass˙1˙L3:
  assumes A1: equiv(A,r) and A2: Congruent(r,f)
  and A3: C ∈ A//r x∈C y∈C
  shows r–f(x)″ = r–f(y)″
⟨*proof*⟩

The values of congruent functions are in the space.

lemma EquivClass˙1˙L4:
  assumes A1: equiv(A,r) and A2: C ∈ A//r x∈C
  and A3: Congruent(r,f)
  shows f(x) ∈ A
⟨*proof*⟩

Equivalence classes are not empty.

lemma EquivClass˙1˙L5:
  assumes A1: refl(A,r) and A2: C ∈ A//r
  shows C≠0
⟨*proof*⟩

To avoid using an axiom of choice, we define the projection using the expression $\bigcup_{x\in C} r(\{f(x)\})$. The next lemma shows that for congruent function this is in the quotient space A/r.

lemma EquivClass˙1˙L6:
  assumes A1: equiv(A,r) and A2: Congruent(r,f)
  and A3: C ∈ A//r
  shows (⋃x∈C. r–f(x)″) ∈ A//r
⟨*proof*⟩

Congruent functions can be projected.

lemma EquivClass˙1˙T0:
  assumes equiv(A,r) Congruent(r,f)
  shows ProjFun(A,r,f) : A//r → A//r
  ⟨*proof*⟩

We now define congruent functions of two variables (binary funtions). The predicate Congruent2 corresponds to congruent2 in Isabelle's standard EquivClass theory, but uses ZF-functions rather than meta-functions.

definition
  Congruent2(r,f) ≡

$(\forall x_1\ x_2\ y_1\ y_2.\ \langle x_1,x_2\rangle \in r \wedge \langle y_1,y_2\rangle \in r \longrightarrow$
$\langle f\langle x_1,y_1\rangle,\ f\langle x_2,y_2\rangle\ \rangle \in r)$

Next we define the notion of projecting a binary operation to the quotient space. This is a very important concept that allows to define quotient groups, among other things.

definition
 ProjFun2(A,r,f) $\equiv$
 $\{\langle p,\bigcup z \in fst(p)\times snd(p).\ r\{f(z)\}\rangle.\ p \in (A//r)\times(A//r)\ \}$

The following lemma is a two-variables equivalent of EquivClass_1_L3.

lemma EquivClass_1_L7:
 assumes A1: equiv(A,r) and A2: Congruent2(r,f)
 and A3: $C_1 \in A//r\ \ C_2 \in A//r$
 and A4: $z_1 \in C_1\times C_2\ \ z_2 \in C_1\times C_2$
 shows $r\{f(z_1)\} = r\{f(z_2)\}$
⟨*proof*⟩

The values of congruent functions of two variables are in the space.

lemma EquivClass_1_L8:
 assumes A1: equiv(A,r) and A2: $C_1 \in A//r$ and A3: $C_2 \in A//r$
 and A4: $z \in C_1\times C_2$ and A5: Congruent2(r,f)
 shows $f(z) \in A$
⟨*proof*⟩

The values of congruent functions are in the space. Note that although this lemma is intended to be used with functions, we don't need to assume that f is a function.

lemma EquivClass_1_L8A:
 assumes A1: equiv(A,r) and A2: $x\in A\ \ y\in A$
 and A3: Congruent2(r,f)
 shows $f\langle x,y\rangle \in A$
⟨*proof*⟩

The following lemma is a two-variables equivalent of EquivClass_1_L6.

lemma EquivClass_1_L9:
 assumes A1: equiv(A,r) and A2: Congruent2(r,f)
 and A3: $p \in (A//r)\times(A//r)$
 shows $(\bigcup z \in fst(p)\times snd(p).\ r\{f(z)\}) \in A//r$
⟨*proof*⟩

Congruent functions of two variables can be projected.

theorem EquivClass_1_T1:
 assumes equiv(A,r) Congruent2(r,f)
 shows ProjFun2(A,r,f) : $(A//r)\times(A//r) \rightarrow A//r$
 ⟨*proof*⟩

The projection diagram commutes. I wish I knew how to draw this diagram in LaTeX.

lemma EquivClass˙1˙L10:
  assumes A1: equiv(A,r) and A2: Congruent2(r,f)
  and A3: x∈A y∈A
  shows ProjFun2(A,r,f)⟨r–x″,r–y″⟩ = r–f⟨x,y⟩″
⟨*proof*⟩

17.2 Projecting commutative, associative and distributive operations.

In this section we show that if the operations are congruent with respect to an equivalence relation then the projection to the quotient space preserves commutativity, associativity and distributivity.

The projection of commutative operation is commutative.

lemma EquivClass˙2˙L1: assumes
  A1: equiv(A,r) and A2: Congruent2(r,f)
  and A3: f –is commutative on″ A
  and A4: c1 ∈ A//r c2 ∈ A//r
  shows ProjFun2(A,r,f)⟨c1,c2⟩ = ProjFun2(A,r,f)⟨c2,c1⟩
⟨*proof*⟩

The projection of commutative operation is commutative.

theorem EquivClass˙2˙T1:
  assumes equiv(A,r) and Congruent2(r,f)
  and f –is commutative on″ A
  shows ProjFun2(A,r,f) –is commutative on″ A//r
  ⟨*proof*⟩

The projection of an associative operation is associative.

lemma EquivClass˙2˙L2:
  assumes A1: equiv(A,r) and A2: Congruent2(r,f)
  and A3: f –is associative on″ A
  and A4: c1 ∈ A//r c2 ∈ A//r c3 ∈ A//r
  and A5: g = ProjFun2(A,r,f)
  shows g⟨g⟨c1,c2⟩,c3⟩ = g⟨c1,g⟨c2,c3⟩⟩
⟨*proof*⟩

The projection of an associative operation is associative on the quotient.

theorem EquivClass˙2˙T2:
  assumes A1: equiv(A,r) and A2: Congruent2(r,f)
  and A3: f –is associative on″ A
  shows ProjFun2(A,r,f) –is associative on″ A//r
⟨*proof*⟩

The essential condition to show that distributivity is preserved by projections to quotient spaces, provided both operations are congruent with respect to the equivalence relation.

lemma EquivClass˙2˙L3:
assumes A1: IsDistributive(X,A,M)
and A2: equiv(X,r)
and A3: Congruent2(r,A) Congruent2(r,M)
and A4: a ∈ X//r b ∈ X//r c ∈ X//r
and A5: A_p = ProjFun2(X,r,A) M_p = ProjFun2(X,r,M)
shows $M_p\langle a,A_p\langle b,c\rangle\rangle = A_p\langle M_p\langle a,b\rangle,M_p\langle a,c\rangle\rangle \wedge$
$M_p\langle A_p\langle b,c\rangle,a\rangle = A_p\langle M_p\langle b,a\rangle, M_p\langle c,a\rangle\rangle$
⟨*proof*⟩

Distributivity is preserved by projections to quotient spaces, provided both operations are congruent with respect to the equivalence relation.

lemma EquivClass˙2˙L4: assumes A1: IsDistributive(X,A,M)
and A2: equiv(X,r)
and A3: Congruent2(r,A) Congruent2(r,M)
shows IsDistributive(X//r,ProjFun2(X,r,A),ProjFun2(X,r,M))
⟨*proof*⟩

17.3 Saturated sets

In this section we consider sets that are saturated with respect to an equivalence relation. A set A is saturated with respect to a relation r if $A = r^{-1}(r(A))$. For equivalence relations saturated sets are unions of equivalence classes. This makes them useful as a tool to define subsets of the quotient space using properties of representants. Namely, we often define a set $B \subseteq X/r$ by saying that $[x]_r \in B$ iff $x \in A$. If A is a saturated set, this definition is consistent in the sense that it does not depend on the choice of x to represent $[x]_r$.

The following defines the notion of a saturated set. Recall that in Isabelle r-(A) is the inverse image of A with respect to relation r. This definition is not specific to equivalence relations.

definition
IsSaturated(r,A) ≡ A = r-(r(A))

For equivalence relations a set is saturated iff it is an image of itself.

lemma EquivClass˙3˙L1: assumes A1: equiv(X,r)
shows IsSaturated(r,A) ⟷ A = r(A)
⟨*proof*⟩

For equivalence relations sets are contained in their images.

lemma EquivClass˙3˙L2: assumes A1: equiv(X,r) and A2: A⊆X
shows A ⊆ r(A)
⟨*proof*⟩

The next lemma shows that if "∼" is an equivalence relation and a set A is such that $a \in A$ and $a \sim b$ implies $b \in A$, then A is saturated with respect to the relation.

lemma EquivClass˙3˙L3: assumes A1: equiv(X,r)
  and A2: r ⊆ X×X and A3: A⊆X
  and A4: ∀x∈A. ∀y∈X. ⟨x,y⟩ ∈ r ⟶ y∈A
  shows IsSaturated(r,A)
⟨*proof*⟩

If $A \subseteq X$ and A is saturated and $x \sim y$, then $x \in A$ iff $y \in A$. Here we show only one direction.

lemma EquivClass˙3˙L4: assumes A1: equiv(X,r)
  and A2: IsSaturated(r,A) and A3: A⊆X
  and A4: ⟨x,y⟩ ∈ r
  and A5: x∈X y∈A
  shows x∈A
⟨*proof*⟩

If $A \subseteq X$ and A is saturated and $x \sim y$, then $x \in A$ iff $y \in A$.

lemma EquivClass˙3˙L5: assumes A1: equiv(X,r)
  and A2: IsSaturated(r,A) and A3: A⊆X
  and A4: x∈X y∈X
  and A5: ⟨x,y⟩ ∈ r
  shows x∈A ⟷ y∈A
⟨*proof*⟩

If A is saturated then $x \in A$ iff its class is in the projection of A.

lemma EquivClass˙3˙L6: assumes A1: equiv(X,r)
  and A2: IsSaturated(r,A) and A3: A⊆X and A4: x∈X
  and A5: B = –r–x″. x∈A″
  shows x∈A ⟷ r–x″ ∈ B
⟨*proof*⟩

A technical lemma involving a projection of a saturated set and a logical epression with exclusive or. Note that we don't really care what Xor is here, this is true for any predicate.

lemma EquivClass˙3˙L7: assumes equiv(X,r)
  and IsSaturated(r,A) and A⊆X
  and x∈X y∈X
  and B = –r–x″. x∈A″
  and (x∈A) Xor (y∈A)
  shows (r–x″ ∈ B) Xor (r–y″ ∈ B)
  ⟨*proof*⟩

end

18 Finite sequences

theory FiniteSeq˙ZF imports Nat˙ZF˙IML func1

begin

This theory treats finite sequences (i.e. maps $n \to X$, where $n = \{0, 1, .., n-1\}$ is a natural number) as lists. It defines and proves the properties of basic operations on lists: concatenation, appending and element etc.

18.1 Lists as finite sequences

A natural way of representing (finite) lists in set theory is through (finite) sequences. In such view a list of elements of a set X is a function that maps the set $\{0, 1, ..n-1\}$ into X. Since natural numbers in set theory are defined so that $n = \{0, 1, ..n-1\}$, a list of length n can be understood as an element of the function space $n \to X$.

We define the set of lists with values in set X as Lists(X).

definition
 Lists(X) ≡ ⋃n∈nat.(n→X)

The set of nonempty X-value listst will be called NELists(X).

definition
 NELists(X) ≡ ⋃n∈nat.(succ(n)→X)

We first define the shift that moves the second sequence to the domain $\{n, .., n + k - 1\}$, where n, k are the lengths of the first and the second sequence, resp. To understand the notation in the definitions below recall that in Isabelle/ZF pred(n) is the previous natural number and denotes the difference between natural numbers n and k.

definition
 ShiftedSeq(b,n) ≡ −⟨j, b(j #- n)⟩. j ∈ NatInterval(n,domain(b))″

We define concatenation of two sequences as the union of the first sequence with the shifted second sequence. The result of concatenating lists a and b is called Concat(a,b).

definition
 Concat(a,b) ≡ a ∪ ShiftedSeq(b,domain(a))

For a finite sequence we define the sequence of all elements except the first one. This corresponds to the "tail" function in Haskell. We call it Tail here as well.

definition
 Tail(a) ≡ −⟨k, a(succ(k))⟩. k ∈ pred(domain(a))″

A dual notion to Tail is the list of all elements of a list except the last one. Borrowing the terminology from Haskell again, we will call this Init.

definition

Init(a) ≡ restrict(a,pred(domain(a)))

Another obvious operation we can talk about is appending an element at the end of a sequence. This is called Append.

definition
Append(a,x) ≡ a ∪ –⟨domain(a),x⟩″

If lists are modeled as finite sequences (i.e. functions on natural intervals $\{0, 1, .., n-1\} = n$) it is easy to get the first element of a list as the value of the sequence at 0. The last element is the value at $n-1$. To hide this behind a familiar name we define the Last element of a list.

definition
Last(a) ≡ a(pred(domain(a)))

Shifted sequence is a function on a the interval of natural numbers.

lemma shifted˙seq˙props:
assumes A1: n ∈ nat k ∈ nat and A2: b:k→X
shows
ShiftedSeq(b,n): NatInterval(n,k) → X
∀ i ∈ NatInterval(n,k). ShiftedSeq(b,n)(i) = b(i #- n)
∀ j∈k. ShiftedSeq(b,n)(n #+ j) = b(j)
⟨*proof*⟩

Basis properties of the contatenation of two finite sequences.

theorem concat˙props:
assumes A1: n ∈ nat k ∈ nat and A2: a:n→X b:k→X
shows
Concat(a,b): n #+ k → X
∀ i∈n. Concat(a,b)(i) = a(i)
∀ i ∈ NatInterval(n,k). Concat(a,b)(i) = b(i #- n)
∀ j ∈ k. Concat(a,b)(n #+ j) = b(j)
⟨*proof*⟩

Properties of concatenating three lists.

lemma concat˙concat˙list:
assumes A1: n ∈ nat k ∈ nat m ∈ nat and
A2: a:n→X b:k→X c:m→X and
A3: d = Concat(Concat(a,b),c)
shows
d : n #+k #+ m → X
∀ j ∈ n. d(j) = a(j)
∀ j ∈ k. d(n #+ j) = b(j)
∀ j ∈ m. d(n #+ k #+ j) = c(j)
⟨*proof*⟩

Properties of concatenating a list with a concatenation of two other lists.

lemma concat˙list˙concat:

assumes A1: n ∈ nat k ∈ nat m ∈ nat and
A2: a:n→X b:k→X c:m→X and
A3: e = Concat(a, Concat(b,c))
shows
e : n #+k #+ m → X
∀ j ∈ n. e(j) = a(j)
∀ j ∈ k. e(n #+ j) = b(j)
∀ j ∈ m. e(n #+ k #+ j) = c(j)
⟨*proof*⟩

Concatenation is associative.

theorem concat_assoc:
assumes A1: n ∈ nat k ∈ nat m ∈ nat and
A2: a:n→X b:k→X c:m→X
shows Concat(Concat(a,b),c) = Concat(a, Concat(b,c))
⟨*proof*⟩

Properties of Tail.

theorem tail_props:
assumes A1: n ∈ nat and A2: a: succ(n) → X
shows
Tail(a) : n → X
∀ k ∈ n. Tail(a)(k) = a(succ(k))
⟨*proof*⟩

Properties of Append. It is a bit surprising that the we don't need to assume that n is a natural number.

theorem append_props:
assumes A1: a: n → X and A2: x∈X and A3: b = Append(a,x)
shows
b : succ(n) → X
∀ k∈n. b(k) = a(k)
b(n) = x
⟨*proof*⟩

A special case of append_props: appending to a nonempty list does not change the head (first element) of the list.

corollary head_of_append:
assumes n∈ nat and a: succ(n) → X and x∈X
shows Append(a,x)(0) = a(0)
⟨*proof*⟩

Tail commutes with Append.

theorem tail_append_commute:
assumes A1: n ∈ nat and A2: a: succ(n) → X and A3: x∈X
shows Append(Tail(a),x) = Tail(Append(a,x))
⟨*proof*⟩

Properties of Init.

theorem init˙props:
  assumes A1: n ∈ nat and A2: a: succ(n) → X
  shows
  Init(a) : n → X
  ∀k∈n. Init(a)(k) = a(k)
  a = Append(Init(a), a(n))
⟨*proof*⟩

If we take init of the result of append, we get back the same list.

lemma init˙append: assumes A1: n ∈ nat and A2: a:n→X and A3: x ∈ X
  shows Init(Append(a,x)) = a
⟨*proof*⟩

A reformulation of definition of Init.

lemma init˙def: assumes n ∈ nat and x:succ(n)→X
  shows Init(x) = restrict(x,n)
  ⟨*proof*⟩

A lemma about extending a finite sequence by one more value. This is just a more explicit version of append˙props.

lemma finseq˙extend:
  assumes a:n→X y∈X b = a ∪ –⟨n,y⟩″
  shows
  b: succ(n) → X
  ∀k∈n. b(k) = a(k)
  b(n) = y
  ⟨*proof*⟩

The next lemma is a bit displaced as it is mainly about finite sets. It is proven here because it uses the notion of Append. Suppose we have a list of element of A is a bijection. Then for every element that does not belong to A we can we can construct a bijection for the set $A \cup \{x\}$ by appending x. This is just a specialised version of lemma bij˙extend˙point from func1.thy.

lemma bij˙append˙point:
  assumes A1: n ∈ nat and A2: b ∈ bij(n,X) and A3: x ∉ X
  shows Append(b,x) ∈ bij(succ(n), X ∪ –x″)
⟨*proof*⟩

The next lemma rephrases the definition of Last. Recall that in ZF we have $\{0, 1, 2, .., n\} = n + 1 =$succ$(n)$.

lemma last˙seq˙elem: assumes a: succ(n) → X shows Last(a) = a(n)
  ⟨*proof*⟩

If two finite sequences are the same when restricted to domain one shorter than the original and have the same value on the last element, then they are equal.

lemma finseq˙restr˙eq: assumes A1: n ∈ nat and
  A2: a: succ(n) → X b: succ(n) → X and
  A3: restrict(a,n) = restrict(b,n) and
  A4: a(n) = b(n)
  shows a = b
⟨*proof*⟩

Concatenating a list of length 1 is the same as appending its first (and only) element. Recall that in ZF set theory $1 = \{0\}$.

lemma append˙1elem: assumes A1: n ∈ nat and
  A2: a: n → X and A3: b : 1 → X
  shows Concat(a,b) = Append(a,b(0))
⟨*proof*⟩

A simple lemma about lists of length 1.

lemma list˙len1˙singleton: assumes A1: x∈X
  shows –⟨0,x⟩" : 1 → X
⟨*proof*⟩

A singleton list is in fact a singleton set with a pair as the only element.

lemma list˙singleton˙pair: assumes A1: x:1→X shows x = –⟨0,x(0)⟩"
⟨*proof*⟩

When we append an element to the empty list we get a list with length 1.

lemma empty˙append1: assumes A1: x∈X
  shows Append(0,x): 1 → X and Append(0,x)(0) = x
⟨*proof*⟩

Appending an element is the same as concatenating with certain pair.

lemma append˙concat˙pair:
  assumes n ∈ nat and a: n → X and x∈X
  shows Append(a,x) = Concat(a,–⟨0,x⟩")
  ⟨*proof*⟩

An associativity property involving concatenation and appending. For proof we just convert appending to concatenation and use concat˙assoc.

lemma concat˙append˙assoc: assumes A1: n ∈ nat k ∈ nat and
  A2: a:n→X   b:k→X and A3: x ∈ X
  shows Append(Concat(a,b),x) = Concat(a, Append(b,x))
⟨*proof*⟩

An identity involving concatenating with init and appending the last element.

lemma concat˙init˙last˙elem:
  assumes n ∈ nat k ∈ nat and
  a: n → X and b : succ(k) → X
  shows Append(Concat(a,Init(b)),b(k)) = Concat(a,b)

⟨*proof*⟩

A lemma about creating lists by composition and how Append behaves in such case.

lemma list˙compose˙append:
  assumes A1: n ∈ nat and A2: a : n → X and
  A3: x ∈ X and A4: c : X → Y
  shows
  c O Append(a,x) : succ(n) → Y
  c O Append(a,x) = Append(c O a, c(x))
⟨*proof*⟩

A lemma about appending an element to a list defined by set comprehension.

lemma set˙list˙append: assumes
  A1: ∀ i ∈ succ(k). b(i) ∈ X and
  A2: a = –⟨i,b(i)⟩. i ∈ succ(k)″
  shows
  a: succ(k) → X
  –⟨i,b(i)⟩. i ∈ k″: k → X
  a = Append(–⟨i,b(i)⟩. i ∈ k″,b(k))
⟨*proof*⟩

An induction theorem for lists.

lemma list˙induct: assumes A1: ∀ b∈1→X. P(b) and
  A2: ∀ b∈NELists(X). P(b) ⟶ (∀ x∈X. P(Append(b,x))) and
  A3: d ∈ NELists(X)
  shows P(d)
⟨*proof*⟩

18.2 Lists and cartesian products

Lists of length n of elements of some set X can be thought of as a model of the cartesian product X^n which is more convenient in many applications.

There is a natural bijection between the space $(n+1) \to X$ of lists of length $n+1$ of elements of X and the cartesian product $(n \to X) \times X$.

lemma lists˙cart˙prod: assumes n ∈ nat
  shows –⟨x,⟨Init(x),x(n)⟩⟩. x ∈ succ(n)→X″ ∈ bij(succ(n)→X,(n→X)×X)
⟨*proof*⟩

We can identify a set X with lists of length one of elements of X.

lemma singleton˙list˙bij: shows –⟨x,x(0)⟩. x∈1→X″ ∈ bij(1→X,X)
⟨*proof*⟩

We can identify a set of X-valued lists of length with X.

lemma list˙singleton˙bij: shows
  –⟨x,–⟨0,x⟩″⟩.x∈X″ ∈ bij(X,1→X) and

–⟨y,y(0)⟩. y∈1→X″ = converse(–⟨x,–⟨0,x⟩″⟩.x∈X″) and
–⟨x,–⟨0,x⟩″⟩.x∈X″ = converse(–⟨y,y(0)⟩. y∈1→X″)
⟨*proof*⟩

What is the inverse image of a set by the natural bijection between X-valued singleton lists and X?

lemma singleton˙vimage: assumes U⊆X shows –x∈1→X. x(0) ∈ U″ = – –⟨0,y⟩″. y∈U″
⟨*proof*⟩

A technical lemma about extending a list by values from a set.

lemma list˙append˙from: assumes A1: n ∈ nat and A2: U ⊆ n→X and A3: V ⊆ X
shows
–x ∈ succ(n)→X. Init(x) ∈ U ∧ x(n) ∈ V″ = (⋃y∈V.–Append(x,y).x∈U″)
⟨*proof*⟩

end

19 Inductive sequences

theory InductiveSeq˙ZF imports Nat˙ZF˙IML FiniteSeq˙ZF

begin

In this theory we discuss sequences defined by conditions of the form $a_0 = x,\ a_{n+1} = f(a_n)$ and similar.

19.1 Sequences defined by induction

One way of defining a sequence (that is a function $a : \mathbb{N} \to X$) is to provide the first element of the sequence and a function to find the next value when we have the current one. This is usually called "defining a sequence by induction". In this section we set up the notion of a sequence defined by induction and prove the theorems needed to use it.

First we define a helper notion of the sequence defined inductively up to a given natural number n.

definition
InductiveSequenceN(x,f,n) ≡
THE a. a: succ(n) → domain(f) ∧ a(0) = x ∧ (∀k∈n. a(succ(k)) = f(a(k)))

From that we define the inductive sequence on the whole set of natural numbers. Recall that in Isabelle/ZF the set of natural numbers is denoted nat.

definition
InductiveSequence(x,f) ≡ ⋃n∈nat. InductiveSequenceN(x,f,n)

First we will consider the question of existence and uniqueness of finite inductive sequences. The proof is by induction and the next lemma is the $P(0)$ step. To understand the notation recall that for natural numbers in set theory we have $n = \{0, 1, .., n - 1\}$ and succ(n)= $\{0, 1, .., n\}$.

lemma indseq˙exun0: assumes A1: f: X→X and A2: x∈X
  shows
  ∃! a. a: succ(0) → X ∧ a(0) = x ∧ (∀k∈0. a(succ(k)) = f(a(k)))
⟨*proof*⟩

A lemma about restricting finite sequences needed for the proof of the inductive step of the existence and uniqueness of finite inductive seqences.

lemma indseq˙restrict:
  assumes A1: f: X→X and A2: x∈X and A3: n ∈ nat and
  A4: a: succ(succ(n))→ X ∧ a(0) = x ∧ (∀k∈succ(n). a(succ(k)) = f(a(k)))
  and A5: a_r = restrict(a,succ(n))
  shows
  a_r: succ(n) → X ∧ a_r(0) = x ∧ (∀k∈n. a_r(succ(k)) = f(a_r(k)))
⟨*proof*⟩

Existence and uniqueness of finite inductive sequences. The proof is by induction and the next lemma is the inductive step.

lemma indseq˙exun˙ind:
  assumes A1: f: X→X and A2: x∈X and A3: n ∈ nat and
  A4: ∃! a. a: succ(n) → X ∧ a(0) = x ∧ (∀k∈n. a(succ(k)) = f(a(k)))
  shows
  ∃! a. a: succ(succ(n)) → X ∧ a(0) = x ∧
  (∀k∈succ(n). a(succ(k)) = f(a(k)))
⟨*proof*⟩

The next lemma combines indseq˙exun0 and indseq˙exun˙ind to show the existence and uniqueness of finite sequences defined by induction.

lemma indseq˙exun:
  assumes A1: f: X→X and A2: x∈X and A3: n ∈ nat
  shows
  ∃! a. a: succ(n) → X ∧ a(0) = x ∧ (∀k∈n. a(succ(k)) = f(a(k)))
⟨*proof*⟩

We are now ready to prove the main theorem about finite inductive sequences.

theorem fin˙indseq˙props:
  assumes A1: f: X→X and A2: x∈X and A3: n ∈ nat and
  A4: a = InductiveSequenceN(x,f,n)
  shows
  a: succ(n) → X
  a(0) = x
  ∀k∈n. a(succ(k)) = f(a(k))
⟨*proof*⟩

A corollary about the domain of a finite inductive sequence.

corollary fin˙indseq˙domain:
  assumes A1: f: X→X and A2: x∈X and A3: n ∈ nat
  shows domain(InductiveSequenceN(x,f,n)) = succ(n)
⟨*proof*⟩

The collection of finite sequences defined by induction is consistent in the sense that the restriction of the sequence defined on a larger set to the smaller set is the same as the sequence defined on the smaller set.

lemma indseq˙consistent: assumes A1: f: X→X and A2: x∈X and
  A3: i ∈ nat j ∈ nat and A4: i ⊆ j
  shows
  restrict(InductiveSequenceN(x,f,j),succ(i)) = InductiveSequenceN(x,f,i)
⟨*proof*⟩

For any two natural numbers one of the corresponding inductive sequences is contained in the other.

lemma indseq˙subsets: assumes A1: f: X→X and A2: x∈X and
  A3: i ∈ nat j ∈ nat and
  A4: a = InductiveSequenceN(x,f,i) b = InductiveSequenceN(x,f,j)
  shows a ⊆ b ∨ b ⊆ a
⟨*proof*⟩

The first theorem about properties of infinite inductive sequences: inductive sequence is a indeed a sequence (i.e. a function on the set of natural numbers.

theorem indseq˙seq: assumes A1: f: X→X and A2: x∈X
  shows InductiveSequence(x,f) : nat → X
⟨*proof*⟩

Restriction of an inductive sequence to a finite domain is the corresponding finite inductive sequence.

lemma indseq˙restr˙eq:
  assumes A1: f: X→X and A2: x∈X and A3: n ∈ nat
  shows
  restrict(InductiveSequence(x,f),succ(n)) = InductiveSequenceN(x,f,n)
⟨*proof*⟩

The first element of the inductive sequence starting at x and generated by f is indeed x.

theorem indseq˙valat0: assumes A1: f: X→X and A2: x∈X
  shows InductiveSequence(x,f)(0) = x
⟨*proof*⟩

An infinite inductive sequence satisfies the inductive relation that defines it.

theorem indseq˙vals:
  assumes A1: f: X→X and A2: x∈X and A3: n ∈ nat

shows
InductiveSequence(x,f)(succ(n)) = f(InductiveSequence(x,f)(n))
⟨*proof*⟩

19.2 Images of inductive sequences

In this section we consider the properties of sets that are images of inductive sequences, that is are of the form $\{f^{(n)}(x) : n \in N\}$ for some x in the domain of f, where $f^{(n)}$ denotes the n'th iteration of the function f. For a function $f : X \to X$ and a point $x \in X$ such set is set is sometimes called the orbit of x generated by f.

The basic properties of orbits.

theorem ind˙seq˙image: assumes A1: f: X→X and A2: x∈X and
A3: A = InductiveSequence(x,f)(nat)
shows x∈A and ∀y∈A. f(y) ∈ A
⟨*proof*⟩

19.3 Subsets generated by a binary operation

In algebra we often talk about sets "generated" by an element, that is sets of the form (in multiplicative notation) $\{a^n | n \in Z\}$. This is a related to a general notion of "power" (as in $a^n = a \cdot a \cdot .. \cdot a$) or multiplicity $n \cdot a = a + a + .. + a$. The intuitive meaning of such notions is obvious, but we need to do some work to be able to use it in the formalized setting. This sections is devoted to sequences that are created by repeatedly applying a binary operation with the second argument fixed to some constant.

Basic propertes of sets generated by binary operations.

theorem binop˙gen˙set:
assumes A1: f: X×Y → X and A2: x∈X y∈Y and
A3: a = InductiveSequence(x,Fix2ndVar(f,y))
shows
a : nat → X
a(nat) ∈ Pow(X)
x ∈ a(nat)
∀z ∈ a(nat). Fix2ndVar(f,y)(z) ∈ a(nat)
⟨*proof*⟩

A simple corollary to the theorem binop˙gen˙set: a set that contains all iterations of the application of a binary operation exists.

lemma binop˙gen˙set˙ex: assumes A1: f: X×Y → X and A2: x∈X y∈Y
shows –A ∈ Pow(X). x∈A ∧ (∀z ∈ A. f⟨z,y⟩ ∈ A) ″ ≠ 0
⟨*proof*⟩

A more general version of binop˙gen˙set where the generating binary operation acts on a larger set.

theorem binop_gen_set1: assumes A1: f: X×Y → X and
  A2: $X_1 \subseteq X$ and A3: $x \in X_1$ $y \in Y$ and
  A4: $\forall t \in X_1.\ f\langle t,y\rangle \in X_1$ and
  A5: a = InductiveSequence(x,Fix2ndVar(restrict(f,X_1×Y),y))
shows
  a : nat → X_1
  a(nat) ∈ Pow(X_1)
  x ∈ a(nat)
  ∀z ∈ a(nat). Fix2ndVar(f,y)(z) ∈ a(nat)
  ∀z ∈ a(nat). f⟨z,y⟩ ∈ a(nat)
⟨*proof*⟩

A generalization of binop_gen_set_ex that applies when the binary operation acts on a larger set. This is used in our Metamath translation to prove the existence of the set of real natural numbers. Metamath defines the real natural numbers as the smallest set that cantains 1 and is closed with respect to operation of adding 1.

lemma binop_gen_set_ex1: assumes A1: f: X×Y → X and
  A2: $X_1 \subseteq X$ and A3: $x \in X_1$ $y \in Y$ and
  A4: $\forall t \in X_1.\ f\langle t,y\rangle \in X_1$
  shows {A ∈ Pow(X_1). x∈A ∧ (∀z ∈ A. f⟨z,y⟩ ∈ A) } ≠ 0
⟨*proof*⟩

19.4 Inductive sequences with changing generating function

A seemingly more general form of a sequence defined by induction is a sequence generated by the difference equation $x_{n+1} = f_n(x_n)$ where $n \mapsto f_n$ is a given sequence of functions such that each maps X into inself. For example when $f_n(x) := x + x_n$ then the equation $S_{n+1} = f_n(S_n)$ describes the sequence $n \mapsto S_n = s_0 + \sum_{i=0}^{n} x_n$, i.e. the sequence of partial sums of the sequence $\{s_0, x_0, x_1, x_3, ..\}$.

The situation where the function that we iterate changes with n can be derived from the simpler case if we define the generating function appropriately. Namely, we replace the generating function in the definitions of InductiveSequenceN by the function $f : X \times n \to X \times n$, $f\langle x, k\rangle = \langle f_k(x), k+1\rangle$ if $k < n$, $\langle f_k(x), k\rangle$ otherwise. The first notion defines the expression we will use to define the generating function. To understand the notation recall that in standard Isabelle/ZF for a pair $s = \langle x, n\rangle$ we have $\mathrm{fst}(s) = x$ and $\mathrm{snd}(s) = n$.

definition
  StateTransfFunNMeta(F,n,s) ≡
  if (snd(s) ∈ n) then ⟨F(snd(s))(fst(s)), succ(snd(s))⟩ else s

Then we define the actual generating function on sets of pairs from $X \times \{0, 1, .., n\}$.

definition

StateTransfFunN(X,F,n) ≡ –⟨s, StateTransfFunNMeta(F,n,s)⟩. s ∈ X×succ(n)″

Having the generating function we can define the expression that we cen use to define the inductive sequence generates.

definition
StatesSeq(x,X,F,n) ≡
InductiveSequenceN(⟨x,0⟩, StateTransfFunN(X,F,n),n)

Finally we can define the sequence given by a initial point x, and a sequence F of n functions.

definition
InductiveSeqVarFN(x,X,F,n) ≡ –⟨k,fst(StatesSeq(x,X,F,n)(k))⟩. k ∈ succ(n)″

The state transformation function (StateTransfFunN is a function that transforms $X \times n$ into itself.

lemma state˙trans˙fun: assumes A1: n ∈ nat and A2: F: n → (X→X)
shows StateTransfFunN(X,F,n): X×succ(n) → X×succ(n)
⟨*proof*⟩

We can apply fin˙indseq˙props to the sequence used in the definition of InductiveSeqVarFN to get the properties of the sequence of states generated by the StateTransfFunN.

lemma states˙seq˙props:
assumes A1: n ∈ nat and A2: F: n → (X→X) and A3: x∈X and
A4: b = StatesSeq(x,X,F,n)
shows
b : succ(n) → X×succ(n)
b(0) = ⟨x,0⟩
∀k ∈ succ(n). snd(b(k)) = k
∀k∈n. b(succ(k)) = ⟨F(k)(fst(b(k))), succ(k)⟩
⟨*proof*⟩

Basic properties of sequences defined by equation $x_{n+1} = f_n(x_n)$.

theorem fin˙indseq˙var˙f˙props:
assumes A1: n ∈ nat and A2: x∈X and A3: F: n → (X→X) and
A4: a = InductiveSeqVarFN(x,X,F,n)
shows
a: succ(n) → X
a(0) = x
∀k∈n. a(succ(k)) = F(k)(a(k))
⟨*proof*⟩

A consistency condition: if we make the sequence of generating functions shorter, then we get a shorter inductive sequence with the same values as in the original sequence.

lemma fin˙indseq˙var˙f˙restrict: assumes
A1: n ∈ nat i ∈ nat x∈X F: n → (X→X) G: i → (X→X)

and A2: i ⊆ n and A3: ∀ j∈i. G(j) = F(j) and A4: k ∈ succ(i)
shows InductiveSeqVarFN(x,X,G,i)(k) = InductiveSeqVarFN(x,X,F,n)(k)
⟨*proof*⟩

end

20 Folding in ZF

theory Fold˙ZF imports InductiveSeq˙ZF

begin

Suppose we have a binary operation $P : X \times X \to X$ written multiplicatively as $P\langle x, y\rangle = x \cdot y$. In informal mathematics we can take a sequence $\{x_k\}_{k\in 0..n}$ of elements of X and consider the product $x_0 \cdot x_1 \cdot .. \cdot x_n$. To do the same thing in formalized mathematics we have to define precisely what is meant by that "$\cdot .. \cdot$". The definitition we want to use is based on the notion of sequence defined by induction discussed in InductiveSeq˙ZF. We don't really want to derive the terminology for this from the word "product" as that would tie it conceptually to the multiplicative notation. This would be awkward when we want to reuse the same notions to talk about sums like $x_0 + x_1 + .. + x_n$. In functional programming there is something called "fold". Namely for a function f, initial point a and list $[b, c, d]$ the expression fold(f, a, [b,c,d]) is defined to be f(f(f(a,b),c),d) (in Haskell something like this is called foldl). If we write f in multiplicative notation we get $a \cdot b \cdot c \cdot d$, so this is exactly what we need. The notion of folds in functional programming is actually much more general that what we need here (not that I know anything about that). In this theory file we just make a slight generalization and talk about folding a list with a binary operation $f : X \times Y \to X$ with X not necessarily the same as Y.

20.1 Folding in ZF

Suppose we have a binary operation $f : X \times Y \to X$. Then every $y \in Y$ defines a transformation of X defined by $T_y(x) = f\langle x, y\rangle$. In IsarMathLib such transformation is called as Fix2ndVar(f,y). Using this notion, given a function $f : X \times Y \to X$ and a sequence $y = \{y_k\}_{k\in N}$ of elements of X we can get a sequence of transformations of X. This is defined in Seq2TransSeq below. Then we use that sequence of tranformations to define the sequence of partial folds (called FoldSeq) by means of InductiveSeqVarFN (defined in InductiveSeq˙ZF theory) which implements the inductive sequence determined by a starting point and a sequence of transformations. Finally, we define the fold of a sequence as the last element of the sequence of the partial folds.

Definition that specifies how to convert a sequence a of elements of Y into a sequence of transformations of X, given a binary operation $f : X \times Y \to X$.

definition
  Seq2TrSeq(f,a) ≡ –⟨k,Fix2ndVar(f,a(k))⟩. k ∈ domain(a)″

Definition of a sequence of partial folds.

definition
  FoldSeq(f,x,a) ≡
  InductiveSeqVarFN(x,fstdom(f),Seq2TrSeq(f,a),domain(a))

Definition of a fold.

definition
  Fold(f,x,a) ≡ Last(FoldSeq(f,x,a))

If X is a set with a binary operation $f : X \times Y \to X$ then Seq2TransSeqN(f,a) converts a sequence a of elements of Y into the sequence of corresponding transformations of X.

lemma seq2trans˙seq˙props:
  assumes A1: n ∈ nat and A2: f : X×Y → X and A3: a:n→Y and
  A4: T = Seq2TrSeq(f,a)
  shows
  T : n → (X→X) and
  ∀k∈n. ∀x∈X. (T(k))(x) = f⟨x,a(k)⟩
⟨*proof*⟩

Basic properties of the sequence of partial folds of a sequence $a = \{y_k\}_{k\in\{0,..,n\}}$.

theorem fold˙seq˙props:
  assumes A1: n ∈ nat and A2: f : X×Y → X and
  A3: y:n→Y and A4: x∈X and A5: Y≠0 and
  A6: F = FoldSeq(f,x,y)
  shows
  F: succ(n) → X
  F(0) = x and
  ∀k∈n. F(succ(k)) = f⟨F(k), y(k)⟩
⟨*proof*⟩

A consistency condition: if we make the list shorter, then we get a shorter sequence of partial folds with the same values as in the original sequence. This can be proven as a special case of fin˙indseq˙var˙f˙restrict but a proof using fold˙seq˙props and induction turns out to be shorter.

lemma foldseq˙restrict: assumes
  n ∈ nat   k ∈ succ(n) and
  i ∈ nat f : X×Y → X a : n → Y b : i → Y and
  n ⊆ i   ∀j ∈ n. b(j) = a(j)   x ∈ X   Y ≠ 0

shows FoldSeq(f,x,b)(k) = FoldSeq(f,x,a)(k)
⟨*proof*⟩

A special case of foldseq˙restrict when the longer sequence is created from the shorter one by appending one element.

corollary fold˙seq˙append:
assumes n ∈ nat   f : X×Y → X   a:n → Y and
x∈X   k ∈ succ(n)   y∈Y
shows FoldSeq(f,x,Append(a,y))(k) = FoldSeq(f,x,a)(k)
⟨*proof*⟩

What we really will be using is the notion of the fold of a sequence, which we define as the last element of (inductively defined) sequence of partial folds. The next theorem lists some properties of the product of the fold operation.

theorem fold˙props:
assumes A1: n ∈ nat and
A2: f : X×Y → X a:n → Y x∈X Y≠0
shows
Fold(f,x,a) = FoldSeq(f,x,a)(n) and
Fold(f,x,a) ∈ X
⟨*proof*⟩

A corner case: what happens when we fold an empty list?

theorem fold˙empty: assumes A1: f : X×Y → X and
A2: a:0→Y x∈X Y≠0
shows Fold(f,x,a) = x
⟨*proof*⟩

The next theorem tells us what happens to the fold of a sequence when we add one more element to it.

theorem fold˙append:
assumes A1: n ∈ nat and A2: f : X×Y → X and
A3: a:n→Y and A4: x∈X and A5: y∈Y
shows
FoldSeq(f,x,Append(a,y))(n) = Fold(f,x,a) and
Fold(f,x,Append(a,y)) = f⟨Fold(f,x,a), y⟩
⟨*proof*⟩

end

21 Partitions of sets

theory Partitions˙ZF imports Finite˙ZF FiniteSeq˙ZF

begin

It is a common trick in proofs that we divide a set into non-overlapping subsets. The first case is when we split the set into two nonempty disjoint sets. Here this is modeled as an ordered pair of sets and the set of such divisions of set X is called Bisections(X). The second variation on this theme is a set-valued function (aren't they all in ZF?) whose values are nonempty and mutually disjoint.

21.1 Bisections

This section is about dividing sets into two non-overlapping subsets.

The set of bisections of a given set A is a set of pairs of nonempty subsets of A that do not overlap and their union is equal to A.

definition
  Bisections(X) = –p ∈ Pow(X)×Pow(X).
  fst(p)≠0 ∧ snd(p)≠0 ∧ fst(p)∩snd(p) = 0 ∧ fst(p)∪snd(p) = X″

Properties of bisections.

lemma bisec˙props: assumes ⟨A,B⟩ ∈ Bisections(X) shows
  A≠0 B≠0 A ⊆ X B ⊆ X A ∩ B = 0 A ∪ B = X X ≠ 0
  ⟨*proof*⟩

Kind of inverse of bisec˙props: a pair of nonempty disjoint sets form a bisection of their union.

lemma is˙bisec:
  assumes A≠0 B≠0 A ∩ B = 0
  shows ⟨A,B⟩ ∈ Bisections(A∪B) ⟨*proof*⟩

Bisection of X is a pair of subsets of X.

lemma bisec˙is˙pair: assumes Q ∈ Bisections(X)
  shows Q = ⟨fst(Q), snd(Q)⟩
  ⟨*proof*⟩

The set of bisections of the empty set is empty.

lemma bisec˙empty: shows Bisections(0) = 0
  ⟨*proof*⟩

The next lemma shows what can we say about bisections of a set with another element added.

lemma bisec˙add˙point:
  assumes A1: x ∉ X and A2: ⟨A,B⟩ ∈ Bisections(X ∪ –x″)
  shows (A = –x″ ∨ B = –x″) ∨ (⟨A - –x″, B - –x″⟩ ∈ Bisections(X))
  ⟨*proof*⟩

A continuation of the lemma bisec˙add˙point that refines the case when the pair with removed point bisects the original set.

lemma bisec˙add˙point˙case3:
assumes A1: $\langle A,B\rangle \in$ Bisections(X $\cup$ –x˝)
and A2: $\langle$A - –x˝, B - –x˝$\rangle \in$ Bisections(X)
shows
($\langle$A, B - –x˝$\rangle \in$ Bisections(X) $\wedge$ x $\in$ B) $\vee$
($\langle$A - –x˝, B$\rangle \in$ Bisections(X) $\wedge$ x $\in$ A)
$\langle$*proof*$\rangle$

Another lemma about bisecting a set with an added point.

lemma point˙set˙bisec:
assumes A1: x $\notin$ X and A2: $\langle$–x˝, A$\rangle \in$ Bisections(X $\cup$ –x˝)
shows A = X and X $\neq$ 0
$\langle$*proof*$\rangle$

Yet another lemma about bisecting a set with an added point, very similar to point˙set˙bisec with almost the same proof.

lemma set˙point˙bisec:
assumes A1: x $\notin$ X and A2: $\langle$A, –x˝$\rangle \in$ Bisections(X $\cup$ –x˝)
shows A = X and X $\neq$ 0
$\langle$*proof*$\rangle$

If a pair of sets bisects a finite set, then both elements of the pair are finite.

lemma bisect˙fin:
assumes A1: A $\in$ FinPow(X) and A2: Q $\in$ Bisections(A)
shows fst(Q) $\in$ FinPow(X) and snd(Q) $\in$ FinPow(X)
$\langle$*proof*$\rangle$

21.2 Partitions

This sections covers the situation when we have an arbitrary number of sets we want to partition into.

We define a notion of a partition as a set valued function such that the values for different arguments are disjoint. The name is derived from the fact that such function ”partitions” the union of its arguments. Please let me know if you have a better idea for a name for such notion. We would prefer to say ”is a partition”, but that reserves the letter ”a” as a keyword(?) which causes problems.

definition
Partition (˙ –is partition˝ [90] 91) where
P –is partition˝ $\equiv$ $\forall$x $\in$ domain(P).
P(x) $\neq$ 0 $\wedge$ ($\forall$y $\in$ domain(P). x$\neq$y $\longrightarrow$ P(x) $\cap$ P(y) = 0)

A fact about lists of mutually disjoint sets.

lemma list˙partition: assumes A1: n $\in$ nat and
A2: a : succ(n) $\rightarrow$ X a –is partition˝
shows ($\bigcup$i$\in$n. a(i)) $\cap$ a(n) = 0

⟨*proof*⟩

We can turn every injection into a partition.

lemma inj˙partition:
  assumes A1: b ∈ inj(X,Y)
  shows
  ∀ x ∈ X. –⟨x, –b(x)″⟩. x ∈ X″(x) = –b(x)″ and
  –⟨x, –b(x)″⟩. x ∈ X″ –is partition″
⟨*proof*⟩

end

22 Enumerations

theory Enumeration˙ZF imports NatOrder˙ZF FiniteSeq˙ZF FinOrd˙ZF

begin

Suppose r is a linear order on a set A that has n elements, where $n \in \mathbb{N}$. In the FinOrd˙ZF theory we prove a theorem stating that there is a unique order isomorphism between $n = \{0, 1, .., n-1\}$ (with natural order) and A. Another way of stating that is that there is a unique way of counting the elements of A in the order increasing according to relation r. Yet another way of stating the same thing is that there is a unique sorted list of elements of A. We will call this list the Enumeration of A.

22.1 Enumerations: definition and notation

In this section we introduce the notion of enumeration and define a proof context (a "locale" in Isabelle terms) that sets up the notation for writing about enumarations.

We define enumeration as the only order isomorphism beween a set A and the number of its elements. We are using the formula $\bigcup\{x\} = x$ to extract the only element from a singleton. Le is the (natural) order on natural numbers, defined is Nat˙ZF theory in the standard Isabelle library.

definition
  Enumeration(A,r) ≡ ⋃ ord˙iso(—A—,Le,A,r)

To set up the notation we define a locale enums. In this locale we will assume that r is a linear order on some set X. In most applications this set will be just the set of natural numbers. Standard Isabelle uses $\leq$ to denote the "less or equal" relation on natural numbers. We will use the $\leq$ symbol to denote the relation r. Those two symbols usually look the same in the presentation, but they are different in the source.To shorten the notation the enumeration Enumeration(A,r) will be denoted as $\sigma(A)$. Similarly as in the Semigroup theory we will write $a \hookleftarrow x$ for the result of appending an element x to the finite sequence (list) a. Finally, $a \sqcup b$ will denote the concatenation of the lists a and b.

locale enums =

fixes X r
assumes linord: IsLinOrder(X,r)

fixes ler (infix ≤ 70)
defines ler˙def[simp]: x ≤ y ≡ ⟨x,y⟩ ∈ r

fixes σ
defines σ˙def [simp]: σ(A) ≡ Enumeration(A,r)

fixes append (infix ↩ 72)
defines append˙def[simp]: a ↩ x ≡ Append(a,x)

fixes concat (infixl ⊔ 69)
defines concat˙def[simp]: a ⊔ b ≡ Concat(a,b)

22.2 Properties of enumerations

In this section we prove basic facts about enumerations.

A special case of the existence and uniqueess of the order isomorphism for finite sets when the first set is a natural number.

lemma (in enums) ord˙iso˙nat˙fin:
assumes A ∈ FinPow(X) and n ∈ nat and A ≈ n
shows ∃!f. f ∈ ord˙iso(n,Le,A,r)
⟨*proof*⟩

An enumeration is an order isomorhism, a bijection, and a list.

lemma (in enums) enum˙props: assumes A ∈ FinPow(X)
shows
σ(A) ∈ ord˙iso(—A—,Le, A,r)
σ(A) ∈ bij(—A—,A)
σ(A) : —A— → A
⟨*proof*⟩

A corollary from enum˙props. Could have been attached as another assertion, but this slows down verification of some other proofs.

lemma (in enums) enum˙fun: assumes A ∈ FinPow(X)
shows σ(A) : —A— → X
⟨*proof*⟩

If a list is an order isomorphism then it must be the enumeration.

lemma (in enums) ord˙iso˙enum: assumes A1: A ∈ FinPow(X) and
A2: n ∈ nat and A3: f ∈ ord˙iso(n,Le,A,r)
shows f = σ(A)
⟨*proof*⟩

What is the enumeration of the empty set?

lemma (in enums) empty˙enum: shows σ(0) = 0
⟨*proof*⟩

Adding a new maximum to a set appends it to the enumeration.

lemma (in enums) enum˙append:
assumes A1: A ∈ FinPow(X) and A2: b ∈ X-A and
A3: ∀ a∈A. a≤b
shows σ(A ∪ –b″) = σ(A)↩ b
⟨*proof*⟩

What is the enumeration of a singleton?

lemma (in enums) enum˙singleton:
assumes A1: x∈X shows σ(–x″): 1 → X and σ(–x″)(0) = x
⟨*proof*⟩

end

23 Semigroups

theory Semigroup˙ZF imports Partitions˙ZF Fold˙ZF Enumeration˙ZF

begin

It seems that the minimal setup needed to talk about a product of a sequence is a set with a binary operation. Such object is called "magma". However, interesting properties show up when the binary operation is associative and such alebraic structure is called a semigroup. In this theory file we define and study sequences of partial products of sequences of magma and semigroup elements.

23.1 Products of sequences of semigroup elements

Semigroup is a a magma in which the binary operation is associative. In this section we mostly study the products of sequences of elements of semigroup. The goal is to establish the fact that taking the product of a sequence is distributive with respect to concatenation of sequences, i.e for two sequences a, b of the semigroup elements we have $\prod(a \sqcup b) = (\prod a) \cdot (\prod b)$, where "$a \sqcup b$" is concatenation of a and b (a++b in Haskell notation). Less formally, we want to show that we can discard parantheses in expressions of the form $(a_0 \cdot a_1 \cdot .. \cdot a_n) \cdot (b_0 \cdot .. \cdot b_k)$.

First we define a notion similar to Fold, except that that the initial element of the fold is given by the first element of sequence. By analogy with Haskell fold we call that Fold1

definition
  Fold1(f,a) ≡ Fold(f,a(0),Tail(a))

The definition of the semigr0 context below introduces notation for writing about finite sequences and semigroup products. In the context we fix the carrier and denote it G. The binary operation on G is called f. All theorems proven in the context semigr0 will implicitly assume that f is an associative operation on G. We will use multiplicative notation for the semigroup operation. The product of a sequence a is denoted $\prod a$. We will write $a \hookleftarrow x$ for the result of appending an element x to the finite sequence (list) a. This is a bit nonstandard, but I don't have a better idea for the "append" notation. Finally, $a \sqcup b$ will denote the concatenation of the lists a and b.

locale semigr0 =

  fixes G f

  assumes assoc˙assum: f –is associative on″ G

  fixes prod (infixl · 72)
  defines prod˙def [simp]: x · y ≡ f⟨x,y⟩

  fixes seqprod (∏ ˙ 71)
  defines seqprod˙def [simp]: ∏ a ≡ Fold1(f,a)

  fixes append (infix ↩ 72)
  defines append˙def [simp]: a ↩ x ≡ Append(a,x)

  fixes concat (infixl ⊔ 69)
  defines concat˙def [simp]: a ⊔ b ≡ Concat(a,b)

The next lemma shows our assumption on the associativity of the semigroup operation in the notation defined in in the semigr0 context.

lemma (in semigr0) semigr˙assoc: assumes x ∈ G y ∈ G z ∈ G
  shows x·y·z = x·(y·z)
  ⟨*proof*⟩

In the way we define associativity the assumption that f is associative on G also implies that it is a binary operation on X.

lemma (in semigr0) semigr˙binop: shows f : G×G → G
  ⟨*proof*⟩

Semigroup operation is closed.

lemma (in semigr0) semigr˙closed:
  assumes a∈G b∈G shows a·b ∈ G
  ⟨*proof*⟩

Lemma append˙1elem written in the notation used in the semigr0 context.

lemma (in semigr0) append˙1elem˙nice:
  assumes n ∈ nat and a: n → X and b : 1 → X
  shows a ⊔ b = a ↩ b(0)
  ⟨*proof*⟩

Lemma concat˙init˙last˙elem rewritten in the notation used in the semigr0 context.

lemma (in semigr0) concat˙init˙last:
  assumes n ∈ nat k ∈ nat and
  a: n → X and b : succ(k) → X
  shows (a ⊔ Init(b)) ↩ b(k) = a ⊔ b
  ⟨*proof*⟩

The product of semigroup (actually, magma – we don't need associativity for this) elements is in the semigroup.

lemma (in semigr0) prod˙type:
  assumes n ∈ nat and a : succ(n) → G
  shows (∏ a) ∈ G
⟨*proof*⟩

What is the product of one element list?

lemma (in semigr0) prod˙of˙1elem: assumes A1: a: 1 → G
  shows (∏ a) = a(0)
⟨*proof*⟩

What happens to the product of a list when we append an element to the list?

lemma (in semigr0) prod˙append: assumes A1: n ∈ nat and
  A2: a : succ(n) → G and A3: x∈G
  shows (∏ a↩x) = (∏ a) · x
⟨*proof*⟩

The main theorem of the section: taking the product of a sequence is distributive with respect to concatenation of sequences. The proof is by induction on the length of the second list.

theorem (in semigr0) prod˙conc˙distr:
  assumes A1: n ∈ nat k ∈ nat and
  A2: a : succ(n) → G   b: succ(k) → G
  shows (∏ a) · (∏ b) = ∏ (a ⊔ b)
⟨*proof*⟩

23.2 Products over sets of indices

In this section we study the properties of expressions of the form $\prod_{i\in\Lambda} a_i = a_{i_0} \cdot a_{i_1} \cdot .. \cdot a_{i-1}$, i.e. what we denote as $\prod(\Lambda,a)$. Λ here is a finite subset of some set X and a is a function defined on X with values in the semigroup G.

Suppose $a : X \to G$ is an indexed family of elements of a semigroup G and $\Lambda = \{i_0, i_1, .., i_{n-1}\} \subseteq \mathbb{N}$ is a finite set of indices. We want to define $\prod_{i\in\Lambda} a_i = a_{i_0} \cdot a_{i_1} \cdot .. \cdot a_{i-1}$. To do that we use the notion of Enumeration defined in the Enumeration˙ZF theory file that takes a set of indices and lists them in increasing order, thus converting it to list. Then we use the Fold1 to multiply the resulting list. Recall that in Isabelle/ZF the capital letter "O" denotes the composition of two functions (or relations).

definition
  SetFold(f,a,Λ,r) = Fold1(f,a O Enumeration(Λ,r))

For a finite subset Λ of a linearly ordered set X we will write $\sigma(\Lambda)$ to denote the enumeration of the elements of Λ, i.e. the only order isomorphism $|\Lambda| \to \Lambda$, where $|\Lambda| \in \mathbb{N}$ is the number of elements of Λ. We also define notation for taking a product over a set of indices of some sequence of semigroup elements. The product of semigroup elements over some set $\Lambda \subseteq X$ of indices of a sequence $a : X \to G$ (i.e. $\prod_{i\in\Lambda} a_i$) is denoted $\prod(\Lambda,a)$. In the semigr1 context we assume that a is a function defined on some linearly ordered set X with values in the semigroup G.

locale semigr1 = semigr0 +

  fixes X r
  assumes linord: IsLinOrder(X,r)

  fixes a
  assumes a˙is˙fun: a : X → G

  fixes σ
  defines σ˙def [simp]: σ(A) ≡ Enumeration(A,r)

  fixes setpr (∏)
  defines setpr˙def [simp]: ∏(Λ,b) ≡ SetFold(f,b,Λ,r)

We can use the enums locale in the semigr0 context.

lemma (in semigr1) enums˙valid˙in˙semigr1: shows enums(X,r)
  ⟨*proof*⟩

Definition of product over a set expressed in notation of the semigr0 locale.

lemma (in semigr1) setproddef:
  shows ∏(Λ,a) = ∏ (a O σ(Λ))

⟨*proof*⟩

A composition of enumeration of a nonempty finite subset of $\mathbb{N}$ with a sequence of elements of G is a nonempty list of elements of G. This implies that a product over set of a finite set of indices belongs to the (carrier of) semigroup.

lemma (in semigr1) setprod˙type: assumes
A1: Λ ∈ FinPow(X) and A2: Λ≠0
shows
∃ n ∈ nat . —Λ— = succ(n) ∧ a O σ(Λ) : succ(n) → G
and ∏(Λ,a) ∈ G
⟨*proof*⟩

The enum˙append lemma from the Enemeration theory specialized for natural numbers.

lemma (in semigr1) semigr1˙enum˙append:
assumes Λ ∈ FinPow(X) and
n ∈ X - Λ and ∀ k∈Λ. ⟨k,n⟩ ∈ r
shows σ(Λ ∪ –n″) = σ(Λ)↩ n
⟨*proof*⟩

What is product over a singleton?

lemma (in semigr1) gen˙prod˙singleton:
assumes A1: x ∈ X
shows ∏(–x″,a) = a(x)
⟨*proof*⟩

A generalization of prod˙append to the products over sets of indices.

lemma (in semigr1) gen˙prod˙append:
assumes
A1: Λ ∈ FinPow(X) and A2: Λ ≠ 0 and
A3: n ∈ X - Λ and
A4: ∀ k∈Λ. ⟨k,n⟩ ∈ r
shows ∏(Λ ∪ –n″, a) = (∏(Λ,a)) · a(n)
⟨*proof*⟩

Very similar to gen˙prod˙append: a relation between a product over a set of indices and the product over the set with the maximum removed.

lemma (in semigr1) gen˙product˙rem˙point:
assumes A1: A ∈ FinPow(X) and
A2: n ∈ A and A4: A - –n″ ≠ 0 and
A3: ∀ k∈A. ⟨k, n⟩ ∈ r
shows
(∏(A - –n″,a)) · a(n) = ∏(A, a)
⟨*proof*⟩

23.3 Commutative semigroups

Commutative semigroups are those whose operation is commutative, i.e. $a \cdot b = b \cdot a$. This implies that for any permutation $s : n \to n$ we have $\prod_{j=0}^{n} a_j = \prod_{j=0}^{n} a_{s(j)}$, or, closer to the notation we are using in the semigr0 context, $\prod a = \prod (a \circ s)$. Maybe one day we will be able to prove this, but for now the goal is to prove something simpler: that if the semigroup operation is commutative taking the product of a sequence is distributive with respect to the operation: $\prod_{j=0}^{n}(a_j \cdot b_j) = \left(\prod_{j=0}^{n} a_j)\right)\left(\prod_{j=0}^{n} b_j)\right)$. Many of the rearrangements (namely those that don't use the inverse) proven in the AbelianGroup˙ZF theory hold in fact in semigroups. Some of them will be reproven in this section.

A rearrangement with 3 elements.

lemma (in semigr0) rearr3elems:
  assumes f –is commutative on˝ G and a∈G b∈G c∈G
  shows a·b·c = a·c·b
  ⟨*proof*⟩

A rearrangement of four elements.

lemma (in semigr0) rearr4elems:
  assumes A1: f –is commutative on˝ G and
  A2: a∈G b∈G c∈G d∈G
  shows a·b·(c·d) = a·c·(b·d)
⟨*proof*⟩

We start with a version of prod˙append that will shorten a bit the proof of the main theorem.

lemma (in semigr0) shorter˙seq: assumes A1: k ∈ nat and
  A2: a ∈ succ(succ(k)) → G
  shows (∏ a) = (∏ Init(a)) · a(succ(k))
⟨*proof*⟩

A lemma useful in the induction step of the main theorem.

lemma (in semigr0) prod˙distr˙ind˙step:
  assumes A1: k ∈ nat and
  A2: a : succ(succ(k)) → G and
  A3: b : succ(succ(k)) → G and
  A4: c : succ(succ(k)) → G and
  A5: ∀j∈succ(succ(k)). c(j) = a(j) · b(j)
  shows
  Init(a) : succ(k) → G
  Init(b) : succ(k) → G
  Init(c) : succ(k) → G
  ∀j∈succ(k). Init(c)(j) = Init(a)(j) · Init(b)(j)
⟨*proof*⟩

For commutative operations taking the product of a sequence is distributive with respect to the operation. This version will probably not be used in applications, it is formulated in a way that is easier to prove by induction. For a more convenient formulation see prod˙comm˙distrib. The proof by induction on the length of the sequence.

theorem (in semigr0) prod˙comm˙distr:
  assumes A1: f –is commutative on″ G and A2: n∈nat
  shows ∀ a b c.
  (a : succ(n)→G ∧ b : succ(n)→G ∧ c : succ(n)→G ∧
  (∀j∈succ(n). c(j) = a(j) · b(j))) ⟶
  (∏ c) = (∏ a) · (∏ b)
⟨*proof*⟩

A reformulation of prod˙comm˙distr that is more convenient in applications.

theorem (in semigr0) prod˙comm˙distrib:
  assumes f –is commutative on″ G and n∈nat and
  a : succ(n)→G b : succ(n)→G c : succ(n)→G and
  ∀j∈succ(n). c(j) = a(j) · b(j)
  shows (∏ c) = (∏ a) · (∏ b)
  ⟨*proof*⟩

A product of two products over disjoint sets of indices is the product over the union.

lemma (in semigr1) prod˙bisect:
  assumes A1: f –is commutative on″ G and A2: Λ ∈ FinPow(X)
  shows
  ∀P ∈ Bisections(Λ). ∏(Λ,a) = (∏(fst(P),a))·(∏(snd(P),a))
⟨*proof*⟩

A better looking reformulation of prod˙bisect.

theorem (in semigr1) prod˙disjoint: assumes
  A1: f –is commutative on″ G and
  A2: A ∈ FinPow(X) A ≠ 0 and
  A3: B ∈ FinPow(X) B ≠ 0 and
  A4: A ∩ B = 0
  shows ∏(A∪B,a) = (∏(A,a))·(∏(B,a))
⟨*proof*⟩

A generalization of prod˙disjoint.

lemma (in semigr1) prod˙list˙of˙lists: assumes
  A1: f –is commutative on″ G and A2: n ∈ nat
  shows ∀M ∈ succ(n) → FinPow(X).
  M –is partition″ ⟶
  (∏ –⟨i,∏(M(i),a)⟩. i ∈ succ(n)″) =
  (∏(⋃i ∈ succ(n). M(i),a))
⟨*proof*⟩

A more convenient reformulation of prod˙list˙of˙lists.

theorem (in semigr1) prod˙list˙of˙sets:
  assumes A1: f –is commutative on˝ G and
  A2: n ∈ nat n ≠ 0 and
  A3: M : n → FinPow(X)   M –is partition˝
  shows
  (∏ –⟨i,∏(M(i),a)⟩. i ∈ n˝) = (∏(⋃i ∈ n. M(i),a))
⟨*proof*⟩

The definition of the product ∏(A,a) ≡ SetFold(f,a,A,r) of a some (finite) set of semigroup elements requires that r is a linear order on the set of indices A. This is necessary so that we know in which order we are multiplying the elements. The product over A is defined so that we have $\prod_A a = \prod a \circ \sigma(A)$ where $\sigma : |A| \rightarrow A$ is the enumeration of A (the only order isomorphism between the number of elements in A and A), see lemma setproddef. However, if the operation is commutative, the order is irrelevant. The next theorem formalizes that fact stating that we can replace the enumeration $\sigma(A)$ by any bijection between $|A|$ and A. In a way this is a generalization of setproddef. The proof is based on application of prod˙list˙of˙sets to the finite collection of singletons that comprise A.

theorem (in semigr1) prod˙order˙irr:
  assumes A1: f –is commutative on˝ G and
  A2: A ∈ FinPow(X) A ≠ 0 and
  A3: b ∈ bij(—A—,A)
  shows (∏ (a O b)) = ∏(A,a)
⟨*proof*⟩

Another way of expressing the fact that the product dos not depend on the order.

corollary (in semigr1) prod˙bij˙same:
  assumes f –is commutative on˝ G and
  A ∈ FinPow(X) A ≠ 0 and
  b ∈ bij(—A—,A) c ∈ bij(—A—,A)
  shows (∏ (a O b)) = (∏ (a O c))
  ⟨*proof*⟩

end

24 Commutative Semigroups

theory CommutativeSemigroup˙ZF imports Semigroup˙ZF

begin

In the Semigroup theory we introduced a notion of SetFold(f,a,Λ,r) that represents the sum of values of some function a valued in a semigroup where the arguments of that function vary over some set Λ. Using the additive notation something like this would be expressed as $\sum_{x\in\Lambda} f(x)$ in informal mathematics. This theory considers an alternative to that notion that is more specific to commutative semigroups.

24.1 Sum of a function over a set

The r parameter in the definition of SetFold(f,a,Λ,r) (from Semigroup˙ZF) represents a linear order relation on Λ that is needed to indicate in what order we are summing the values $f(x)$. If the semigroup operation is commutative the order does not matter and the relation r is not needed. In this section we define a notion of summing up values of some function $a : X \to G$ over a finite set of indices $\Gamma \subseteq X$, without using any order relation on X.

We define the sum of values of a function $a : X \to G$ over a set Λ as the only element of the set of sums of lists that are bijections between the number of values in Λ (which is a natural number $n = \{0, 1, .., n-1\}$ if Λ is finite) and Λ. The notion of Fold1(f,c) is defined in Semigroup˙ZF as the fold (sum) of the list c starting from the first element of that list. The intention is to use the fact that since the result of summing up a list does not depend on the order, the set –Fold1(f,a O b). b ∈ bij(—Λ—, Λ)˝ is a singleton and we can extract its only value by taking its union.

definition
  CommSetFold(f,a,Λ) = $\bigcup$–Fold1(f,a O b). b ∈ bij(—Λ—, Λ)˝

the next locale sets up notation for writing about summation in commutative semigroups. We define two kinds of sums. One is the sum of elements of a list (which are just functions defined on a natural number) and the second one represents a more general notion the sum of values of a semigroup valued function over some set of arguments. Since those two types of sums are different notions they are represented by different symbols. However in the presentations they are both intended to be printed as $\sum$.

locale commsemigr =

  fixes G f

  assumes csgassoc: f –is associative on˝ G

  assumes csgcomm: f –is commutative on˝ G

  fixes csgsum (infixl + 69)
  defines csgsum˙def[simp]: x + y ≡ f⟨x,y⟩

  fixes X a
  assumes csgaisfun: a : X → G

  fixes csglistsum ($\sum$ ˙ 70)
  defines csglistsum˙def[simp]: $\sum$ k ≡ Fold1(f,k)

fixes csgsetsum ($\sum$)
defines csgsetsum˙def[simp]: $\sum$(A,h) $\equiv$ CommSetFold(f,h,A)

Definition of a sum of function over a set in notation defined in the commsemigr locale.

lemma (in commsemigr) CommSetFolddef:
shows ($\sum$(A,a)) = ($\bigcup$–$\sum$ (a O b). b $\in$ bij(—A—, A)")
⟨*proof*⟩

The next lemma states that the result of a sum does not depend on the order we calculate it. This is similar to lemma prod˙order˙irr in the Semigroup theory, except that the semigr1 locale assumes that the domain of the function we sum up is linearly ordered, while in commsemigr we don't have this assumption.

lemma (in commsemigr) sum˙over˙set˙bij:
assumes A1: A $\in$ FinPow(X) A $\neq$ 0 and A2: b $\in$ bij(—A—,A)
shows ($\sum$(A,a)) = ($\sum$ (a O b))
⟨*proof*⟩

The result of a sum is in the semigroup. Also, as the second assertion we show that every semigroup valued function generates a homomorphism between the finite subsets of a semigroup and the semigroup. Adding an element to a set coresponds to adding a value.

lemma (in commsemigr) sum˙over˙set˙add˙point:
assumes A1: A $\in$ FinPow(X) A $\neq$ 0
shows $\sum$(A,a) $\in$ G and
$\forall$x $\in$ X-A. $\sum$(A $\cup$ –x",a) = ($\sum$(A,a)) + a(x)
⟨*proof*⟩

end

25 Monoids

theory Monoid˙ZF imports func˙ZF

begin

This theory provides basic facts about monoids.

25.1 Definition and basic properties

In this section we talk about monoids. The notion of a monoid is similar to the notion of a semigroup except that we require the existence of a neutral element. It is also similar to the notion of group except that we don't require existence of the inverse.

Monoid is a set G with an associative operation and a neutral element. The operation is a function on $G \times G$ with values in G. In the context of ZF set theory this means that it is a set of pairs $\langle x, y \rangle$, where $x \in G \times G$ and $y \in G$. In other words the operation is a certain subset of $(G \times G) \times G$. We express all this by defing a predicate IsAmonoid(G,f). Here G is the "carrier" of the group and f is the binary operation on it.

definition
 IsAmonoid(G,f) ≡
 f –is associative on″ G ∧
 (∃e∈G. (∀ g∈G. ((f(⟨e,g⟩) = g) ∧ (f(⟨g,e⟩) = g))))

The next locale called "monoid0" defines a context for theorems that concern monoids. In this contex we assume that the pair (G, f) is a monoid. We will use the ⊕ symbol to denote the monoid operation (for no particular reason).

locale monoid0 =
 fixes G
 fixes f
 assumes monoidAsssum: IsAmonoid(G,f)

 fixes monoper (infixl ⊕ 70)
 defines monoper˙def [simp]: a ⊕ b ≡ f⟨a,b⟩

The result of the monoid operation is in the monoid (carrier).

lemma (in monoid0) group0˙1˙L1:
 assumes a∈G b∈G shows a⊕b ∈ G
 ⟨*proof*⟩

There is only one neutral element in a monoid.

lemma (in monoid0) group0˙1˙L2: shows
 ∃!e. e∈G ∧ (∀ g∈G. ((e⊕g = g) ∧ g⊕e = g))
⟨*proof*⟩

We could put the definition of neutral element anywhere, but it is only usable in conjuction with the above lemma.

definition
 TheNeutralElement(G,f) ≡
 (THE e. e∈G ∧ (∀ g∈G. f⟨e,g⟩ = g ∧ f⟨g,e⟩ = g))

The neutral element is neutral.

lemma (in monoid0) unit˙is˙neutral:
 assumes A1: e = TheNeutralElement(G,f)
 shows e ∈ G ∧ (∀g∈G. e ⊕ g = g ∧ g ⊕ e = g)
⟨*proof*⟩

The monoid carrier is not empty.

lemma (in monoid0) group0˙1˙L3A: shows G≠0

⟨*proof*⟩

The range of the monoid operation is the whole monoid carrier.

lemma (in monoid0) group0˙1˙L3B: shows range(f) = G
⟨*proof*⟩

Another way to state that the range of the monoid operation is the whole monoid carrier.

lemma (in monoid0) range˙carr: shows f(G×G) = G
⟨*proof*⟩

In a monoid any neutral element is the neutral element.

lemma (in monoid0) group0˙1˙L4:
assumes A1: e ∈ G ∧ (∀g∈G. e ⊕ g = g ∧ g ⊕ e = g)
shows e = TheNeutralElement(G,f)
⟨*proof*⟩

The next lemma shows that if the if we restrict the monoid operation to a subset of G that contains the neutral element, then the neutral element of the monoid operation is also neutral with the restricted operation.

lemma (in monoid0) group0˙1˙L5:
assumes A1: ∀x∈H.∀y∈H. x⊕y ∈ H
and A2: H⊆G
and A3: e = TheNeutralElement(G,f)
and A4: g = restrict(f,H×H)
and A5: e∈H
and A6: h∈H
shows g⟨e,h⟩ = h ∧ g⟨h,e⟩ = h
⟨*proof*⟩

The next theorem shows that if the monoid operation is closed on a subset of G then this set is a (sub)monoid (although we do not define this notion). This fact will be useful when we study subgroups.

theorem (in monoid0) group0˙1˙T1:
assumes A1: H –is closed under″ f
and A2: H⊆G
and A3: TheNeutralElement(G,f) ∈ H
shows IsAmonoid(H,restrict(f,H×H))
⟨*proof*⟩

Under the assumptions of group0˙1˙T1 the neutral element of a submonoid is the same as that of the monoid.

lemma group0˙1˙L6:
assumes A1: IsAmonoid(G,f)
and A2: H –is closed under″ f
and A3: H⊆G
and A4: TheNeutralElement(G,f) ∈ H

shows TheNeutralElement(H,restrict(f,H×H)) = TheNeutralElement(G,f)
⟨*proof*⟩

If a sum of two elements is not zero, then at least one has to be nonzero.

lemma (in monoid0) sum˙nonzero˙elmnt˙nonzero:
assumes a ⊕ b ≠ TheNeutralElement(G,f)
shows a ≠ TheNeutralElement(G,f) ∨ b ≠ TheNeutralElement(G,f)
⟨*proof*⟩

end

26 Groups - introduction

theory Group˙ZF imports Monoid˙ZF

begin

This theory file covers basics of group theory.

26.1 Definition and basic properties of groups

In this section we define the notion of a group and set up the notation for discussing groups. We prove some basic theorems about groups.

To define a group we take a monoid and add a requirement that the right inverse needs to exist for every element of the group.

definition
IsAgroup(G,f) ≡
(IsAmonoid(G,f) ∧ (∀ g∈G. ∃ b∈G. f⟨g,b⟩ = TheNeutralElement(G,f)))

We define the group inverse as the set $\{\langle x,y\rangle \in G\times G : x\cdot y = e\}$, where e is the neutral element of the group. This set (which can be written as $(\cdot)^{-1}\{e\}$) is a certain relation on the group (carrier). Since, as we show later, for every $x\in G$ there is exactly one $y\in G$ such that $x\cdot y = e$ this relation is in fact a function from G to G.

definition
GroupInv(G,f) ≡ –⟨x,y⟩ ∈ G×G. f⟨x,y⟩ = TheNeutralElement(G,f)˝

We will use the miltiplicative notation for groups. The neutral element is denoted 1.

locale group0 =
fixes G
fixes P
assumes groupAssum: IsAgroup(G,P)

fixes neut (**1**)

defines neut˙def[simp]: $\mathbf{1} \equiv$ TheNeutralElement(G,P)

fixes groper (infixl · 70)
defines groper˙def[simp]: a · b $\equiv$ P⟨a,b⟩

fixes inv (˙$^{-1}$ [90] 91)
defines inv˙def[simp]: x^{-1} $\equiv$ GroupInv(G,P)(x)

First we show a lemma that says that we can use theorems proven in the monoid0 context (locale).

lemma (in group0) group0˙2˙L1: shows monoid0(G,P)
⟨*proof*⟩

In some strange cases Isabelle has difficulties with applying the definition of a group. The next lemma defines a rule to be applied in such cases.

lemma definition˙of˙group: assumes IsAmonoid(G,f)
and ∀g∈G. ∃b∈G. f⟨g,b⟩ = TheNeutralElement(G,f)
shows IsAgroup(G,f)
⟨*proof*⟩

A technical lemma that allows to use 1 as the neutral element of the group without referencing a list of lemmas and definitions.

lemma (in group0) group0˙2˙L2:
shows $\mathbf{1}$∈G ∧ (∀g∈G.($\mathbf{1}$·g = g ∧ g·$\mathbf{1}$ = g))
⟨*proof*⟩

The group is closed under the group operation. Used all the time, useful to have handy.

lemma (in group0) group˙op˙closed: assumes a∈G b∈G
shows a·b ∈ G ⟨*proof*⟩

The group operation is associative. This is another technical lemma that allows to shorten the list of referenced lemmas in some proofs.

lemma (in group0) group˙oper˙assoc:
assumes a∈G b∈G c∈G shows a·(b·c) = a·b·c
⟨*proof*⟩

The group operation maps $G \times G$ into G. It is conveniet to have this fact easily accessible in the group0 context.

lemma (in group0) group˙oper˙assocA: shows P : G×G→G
⟨*proof*⟩

The definition of a group requires the existence of the right inverse. We show that this is also the left inverse.

theorem (in group0) group0˙2˙T1:
assumes A1: g∈G and A2: b∈G and A3: g·b = $\mathbf{1}$
shows b·g = $\mathbf{1}$

⟨proof⟩

For every element of a group there is only one inverse.

lemma (in group0) group0˙2˙L4:
  assumes A1: x∈G shows ∃!y. y∈G ∧ x·y = **1**
⟨proof⟩

The group inverse is a function that maps G into G.

theorem group0˙2˙T2:
  assumes A1: IsAgroup(G,f) shows GroupInv(G,f) : G→G
⟨proof⟩

We can think about the group inverse (the function) as the inverse image of the neutral element. Recall that in Isabelle f-(A) denotes the inverse image of the set A.

theorem (in group0) group0˙2˙T3: shows P-–**1**″ = GroupInv(G,P)
⟨proof⟩

The inverse is in the group.

lemma (in group0) inverse˙in˙group: assumes A1: x∈G shows x^{-1}∈G
⟨proof⟩

The notation for the inverse means what it is supposed to mean.

lemma (in group0) group0˙2˙L6:
  assumes A1: x∈G shows x·x^{-1} = **1** ∧ x^{-1}·x = **1**
⟨proof⟩

The next two lemmas state that unless we multiply by the neutral element, the result is always different than any of the operands.

lemma (in group0) group0˙2˙L7:
  assumes A1: a∈G and A2: b∈G and A3: a·b = a
  shows b=**1**
⟨proof⟩

See the comment to group0˙2˙L7.

lemma (in group0) group0˙2˙L8:
  assumes A1: a∈G and A2: b∈G and A3: a·b = b
  shows a=**1**
⟨proof⟩

The inverse of the neutral element is the neutral element.

lemma (in group0) group˙inv˙of˙one: shows $\mathbf{1}^{-1}$ = **1**
  ⟨proof⟩

if $a^{-1} = 1$, then $a = 1$.

lemma (in group0) group0˙2˙L8A:

assumes A1: a∈G and A2: $a^{-1} = \mathbf{1}$
shows $a = \mathbf{1}$
⟨*proof*⟩

If a is not a unit, then its inverse is not a unit either.

lemma (in group0) group0_2_L8B:
assumes a∈G and $a \neq \mathbf{1}$
shows $a^{-1} \neq \mathbf{1}$ ⟨*proof*⟩

If a^{-1} is not a unit, then a is not a unit either.

lemma (in group0) group0_2_L8C:
assumes a∈G and $a^{-1} \neq \mathbf{1}$
shows $a \neq \mathbf{1}$
⟨*proof*⟩

If a product of two elements of a group is equal to the neutral element then they are inverses of each other.

lemma (in group0) group0_2_L9:
assumes A1: a∈G and A2: b∈G and A3: $a \cdot b = \mathbf{1}$
shows $a = b^{-1}$ and $b = a^{-1}$
⟨*proof*⟩

It happens quite often that we know what is (have a meta-function for) the right inverse in a group. The next lemma shows that the value of the group inverse (function) is equal to the right inverse (meta-function).

lemma (in group0) group0_2_L9A:
assumes A1: $\forall g \in G.\ b(g) \in G \wedge g \cdot b(g) = \mathbf{1}$
shows $\forall g \in G.\ b(g) = g^{-1}$
⟨*proof*⟩

What is the inverse of a product?

lemma (in group0) group_inv_of_two:
assumes A1: a∈G and A2: b∈G
shows $b^{-1} \cdot a^{-1} = (a \cdot b)^{-1}$
⟨*proof*⟩

What is the inverse of a product of three elements?

lemma (in group0) group_inv_of_three:
assumes A1: a∈G b∈G c∈G
shows
$(a \cdot b \cdot c)^{-1} = c^{-1} \cdot (a \cdot b)^{-1}$
$(a \cdot b \cdot c)^{-1} = c^{-1} \cdot (b^{-1} \cdot a^{-1})$
$(a \cdot b \cdot c)^{-1} = c^{-1} \cdot b^{-1} \cdot a^{-1}$
⟨*proof*⟩

The inverse of the inverse is the element.

lemma (in group0) group_inv_of_inv:

assumes a∈G shows a = $(a^{-1})^{-1}$
⟨*proof*⟩

Group inverse is nilpotent, therefore a bijection and involution.

lemma (in group0) group˙inv˙bij:
shows GroupInv(G,P) O GroupInv(G,P) = id(G) and GroupInv(G,P) ∈ bij(G,G) and
GroupInv(G,P) = converse(GroupInv(G,P))
⟨*proof*⟩

A set comprehension form of the image of a set under the group inverse.

lemma (in group0) ginv˙image: assumes V⊆G
shows GroupInv(G,P)(V) ⊆ G and GroupInv(G,P)(V) = –g^{-1}. g ∈ V″
⟨*proof*⟩

Inverse of an element that belongs to the inverse of the set belongs to the set.

lemma (in group0) ginv˙image˙el: assumes V⊆G g ∈ GroupInv(G,P)(V)
shows g^{-1} ∈ V
⟨*proof*⟩

For the group inverse the image is the same as inverse image.

lemma (in group0) inv˙image˙vimage: shows GroupInv(G,P)(V) = GroupInv(G,P)-(V)
⟨*proof*⟩

If the unit is in a set then it is in the inverse of that set.

lemma (in group0) neut˙inv˙neut: assumes A⊆G and **1**∈A
shows **1** ∈ GroupInv(G,P)(A)
⟨*proof*⟩

The group inverse is onto.

lemma (in group0) group˙inv˙surj: shows GroupInv(G,P)(G) = G
⟨*proof*⟩

If $a^{-1} \cdot b = 1$, then $a = b$.

lemma (in group0) group0˙2˙L11:
assumes A1: a∈G b∈G and A2: a^{-1}·b = **1**
shows a=b
⟨*proof*⟩

If $a \cdot b^{-1} = 1$, then $a = b$.

lemma (in group0) group0˙2˙L11A:
assumes A1: a∈G b∈G and A2: a·b^{-1} = **1**
shows a=b
⟨*proof*⟩

If if the inverse of b is different than a, then the inverse of a is different than b.

lemma (in group0) group0˙2˙L11B:
  assumes A1: a∈G and A2: $b^{-1} \neq a$
  shows $a^{-1} \neq b$
⟨*proof*⟩

What is the inverse of ab^{-1} ?

lemma (in group0) group0˙2˙L12:
  assumes A1: a∈G b∈G
  shows
  $(a\cdot b^{-1})^{-1} = b\cdot a^{-1}$
  $(a^{-1}\cdot b)^{-1} = b^{-1}\cdot a$
⟨*proof*⟩

A couple useful rearrangements with three elements: we can insert a $b \cdot b^{-1}$ between two group elements (another version) and one about a product of an element and inverse of a product, and two others.

lemma (in group0) group0˙2˙L14A:
  assumes A1: a∈G b∈G c∈G
  shows
  $a\cdot c^{-1}= (a\cdot b^{-1})\cdot(b\cdot c^{-1})$
  $a^{-1}\cdot c = (a^{-1}\cdot b)\cdot(b^{-1}\cdot c)$
  $a\cdot(b\cdot c)^{-1} = a\cdot c^{-1}\cdot b^{-1}$
  $a\cdot(b\cdot c^{-1}) = a\cdot b\cdot c^{-1}$
  $(a\cdot b^{-1}\cdot c^{-1})^{-1} = c\cdot b\cdot a^{-1}$
  $a\cdot b\cdot c^{-1}\cdot(c\cdot b^{-1}) = a$
  $a\cdot(b\cdot c)\cdot c^{-1} = a\cdot b$
⟨*proof*⟩

A simple equation to solve

lemma (in group0) simple˙equation0:
  assumes a∈G b∈G c∈G $a\cdot b^{-1} = c^{-1}$
  shows $c = b\cdot a^{-1}$
⟨*proof*⟩

Another simple equation

lemma (in group0) simple˙equation1:
  assumes a∈G b∈G c∈G $a^{-1}\cdot b = c^{-1}$
  shows $c = b^{-1}\cdot a$
⟨*proof*⟩

Another lemma about rearranging a product of four group elements.

lemma (in group0) group0˙2˙L15:
  assumes A1: a∈G b∈G c∈G d∈G
  shows $(a\cdot b)\cdot(c\cdot d)^{-1} = a\cdot(b\cdot d^{-1})\cdot a^{-1}\cdot(a\cdot c^{-1})$
⟨*proof*⟩

We can cancel an element with its inverse that is written next to it.

lemma (in group0) inv˙cancel˙two:

assumes A1: a∈G b∈G
shows
$a \cdot b^{-1} \cdot b = a$
$a \cdot b \cdot b^{-1} = a$
$a^{-1} \cdot (a \cdot b) = b$
$a \cdot (a^{-1} \cdot b) = b$
⟨*proof*⟩

Another lemma about cancelling with two group elements.

lemma (in group0) group0˙2˙L16A:
assumes A1: a∈G b∈G
shows $a \cdot (b \cdot a)^{-1} = b^{-1}$
⟨*proof*⟩

Some other identities with three element and cancelling.

lemma (in group0) cancel˙middle:
assumes a∈G b∈G c∈G
shows
$(a \cdot b)^{-1} \cdot (a \cdot c) = b^{-1} \cdot c$
$(a \cdot b) \cdot (c \cdot b)^{-1} = a \cdot c^{-1}$
$a^{-1} \cdot (a \cdot b \cdot c) \cdot c^{-1} = b$
$a \cdot (b \cdot c^{-1}) \cdot c = a \cdot b$
$a \cdot b^{-1} \cdot (b \cdot c^{-1}) = a \cdot c^{-1}$
⟨*proof*⟩

Adding a neutral element to a set that is closed under the group operation results in a set that is closed under the group operation.

lemma (in group0) group0˙2˙L17:
assumes H⊆G
and H –is closed under˝ P
shows (H ∪ –**1**˝) –is closed under˝ P
⟨*proof*⟩

We can put an element on the other side of an equation.

lemma (in group0) group0˙2˙L18:
assumes A1: a∈G b∈G
and A2: $c = a \cdot b$
shows $c \cdot b^{-1} = a$ $a^{-1} \cdot c = b$
⟨*proof*⟩

We can cancel an element on the right from both sides of an equation.

lemma (in group0) cancel˙right: assumes a∈G b∈G c∈G $a \cdot b = c \cdot b$
shows a = c
⟨*proof*⟩

We can cancel an element on the left from both sides of an equation.

lemma (in group0) cancel˙left: assumes a∈G b∈G c∈G $a \cdot b = a \cdot c$
shows b=c

⟨*proof*⟩

Multiplying different group elements by the same factor results in different group elements.

lemma (in group0) group0˙2˙L19:
  assumes A1: a∈G b∈G c∈G and A2: a≠b
  shows a·c ≠ b·c and c·a ≠ c·b
⟨*proof*⟩

26.2 Subgroups

There are two common ways to define subgroups. One requires that the group operation is closed in the subgroup. The second one defines subgroup as a subset of a group which is itself a group under the group operations. We use the second approach because it results in shorter definition.

The rest of this section is devoted to proving the equivalence of these two definitions of the notion of a subgroup.

A pair (H,P) is a subgroup if H forms a group with the operation P restricted to $H \times H$. It may be surprising that we don't require H to be a subset of G. This however can be inferred from the definition if the pair (G,P) is a group, see lemma group0˙3˙L2.

definition
  IsAsubgroup(H,P) ≡ IsAgroup(H, restrict(P,H×H))

Formally the group operation in a subgroup is different than in the group as they have different domains. Of course we want to use the original operation with the associated notation in the subgroup. The next couple of lemmas will allow for that.

The next lemma states that the neutral element of a subgroup is in the subgroup and it is both right and left neutral there. The notation is very ugly because we don't want to introduce a separate notation for the subgroup operation.

lemma group0˙3˙L1:
  assumes A1: IsAsubgroup(H,f)
  and A2: n = TheNeutralElement(H,restrict(f,H×H))
  shows n ∈ H
  ∀h∈H. restrict(f,H×H)⟨n,h ⟩ = h
  ∀h∈H. restrict(f,H×H)⟨h,n⟩ = h
⟨*proof*⟩

A subgroup is contained in the group.

lemma (in group0) group0˙3˙L2:
  assumes A1: IsAsubgroup(H,P)
  shows H ⊆ G
⟨*proof*⟩

The group's neutral element (denoted 1 in the group0 context) is a neutral element for the subgroup with respect to the group action.

lemma (in group0) group0˙3˙L3:
  assumes IsAsubgroup(H,P)
  shows ∀h∈H. **1**·h = h ∧ h·**1** = h
  ⟨*proof*⟩

The neutral element of a subgroup is the same as that of the group.

lemma (in group0) group0˙3˙L4: assumes A1: IsAsubgroup(H,P)
  shows TheNeutralElement(H,restrict(P,H×H)) = **1**
⟨*proof*⟩

The neutral element of the group (denoted 1 in the group0 context) belongs to every subgroup.

lemma (in group0) group0˙3˙L5: assumes A1: IsAsubgroup(H,P)
  shows **1** ∈ H
⟨*proof*⟩

Subgroups are closed with respect to the group operation.

lemma (in group0) group0˙3˙L6: assumes A1: IsAsubgroup(H,P)
  and A2: a∈H b∈H
  shows a·b ∈ H
⟨*proof*⟩

A preliminary lemma that we need to show that taking the inverse in the subgroup is the same as taking the inverse in the group.

lemma group0˙3˙L7A:
  assumes A1: IsAgroup(G,f)
  and A2: IsAsubgroup(H,f) and A3: g = restrict(f,H×H)
  shows GroupInv(G,f) ∩ H×H = GroupInv(H,g)
⟨*proof*⟩

Using the lemma above we can show the actual statement: taking the inverse in the subgroup is the same as taking the inverse in the group.

theorem (in group0) group0˙3˙T1:
  assumes A1: IsAsubgroup(H,P)
  and A2: g = restrict(P,H×H)
  shows GroupInv(H,g) = restrict(GroupInv(G,P),H)
⟨*proof*⟩

A sligtly weaker, but more convenient in applications, reformulation of the above theorem.

theorem (in group0) group0˙3˙T2:
  assumes IsAsubgroup(H,P)
  and g = restrict(P,H×H)
  shows ∀h∈H. GroupInv(H,g)(h) = h^{-1}
  ⟨*proof*⟩

Subgroups are closed with respect to taking the group inverse.

theorem (in group0) group0_3_T3A:
  assumes A1: IsAsubgroup(H,P) and A2: $h \in H$
  shows $h^{-1} \in H$
⟨*proof*⟩

The next theorem states that a nonempty subset of a group G that is closed under the group operation and taking the inverse is a subgroup of the group.

theorem (in group0) group0_3_T3:
  assumes A1: $H \neq 0$
  and A2: $H \subseteq G$
  and A3: H {is closed under} P
  and A4: $\forall x \in H.\ x^{-1} \in H$
  shows IsAsubgroup(H,P)
⟨*proof*⟩

Intersection of subgroups is a subgroup. This lemma is obsolete and should be replaced by subgroup_inter.

lemma group0_3_L7:
  assumes A1: IsAgroup(G,f)
  and A2: IsAsubgroup(H_1,f)
  and A3: IsAsubgroup(H_2,f)
  shows IsAsubgroup($H_1 \cap H_2$,restrict(f,$H_1 \times H_1$))
⟨*proof*⟩

Intersection of subgroups is a subgroup.

lemma (in group0) subgroup_inter: assumes IsAsubgroup(H_1,P) and IsAsubgroup(H_2,P)
  shows IsAsubgroup($H_1 \cap H_2$,P)
⟨*proof*⟩

The range of the subgroup operation is the whole subgroup.

lemma image_subgr_op: assumes A1: IsAsubgroup(H,P)
  shows restrict(P,$H \times H$)($H \times H$) = H
⟨*proof*⟩

If we restrict the inverse to a subgroup, then the restricted inverse is onto the subgroup.

lemma (in group0) restr_inv_onto: assumes A1: IsAsubgroup(H,P)
  shows restrict(GroupInv(G,P),H)(H) = H
⟨*proof*⟩

A union of two subgroups is a subgroup iff one of the subgroups is a subset of the other subgroup.

lemma (in group0) union_subgroups:
  assumes IsAsubgroup(H_1,P) and IsAsubgroup(H_2,P)
  shows IsAsubgroup($H_1 \cup H_2$,P) $\longleftrightarrow$ ($H_1 \subseteq H_2 \vee H_2 \subseteq H_1$)
⟨*proof*⟩

Transitivity for "is a subgroup of" relation. The proof (probably) uses the lemma restrict˙restrict from standard Isabelle/ZF library which states that restrict(restrict(f,A),B) = restrict(f,A∩B). That lemma is added to the simplifier, so it does not have to be referenced explicitly in the proof below.

lemma subgroup˙transitive:
 assumes IsAgroup(G_3,P) IsAsubgroup(G_2,P) IsAsubgroup(G_1,restrict(P,G_2×G_2))
 shows IsAsubgroup(G_1,P)
⟨*proof*⟩

end

27 Groups 1

theory Group˙ZF˙1 imports Group˙ZF

begin

In this theory we consider right and left translations and odd functions.

27.1 Translations

In this section we consider translations. Translations are maps $T : G \to G$ of the form $T_g(a) = g \cdot a$ or $T_g(a) = a \cdot g$. We also consider two-dimensional translations $T_g : G \times G \to G \times G$, where $T_g(a, b) = (a \cdot g, b \cdot g)$ or $T_g(a, b) = (g \cdot a, g \cdot b)$.

For an element $a \in G$ the right translation is defined a function (set of pairs) such that its value (the second element of a pair) is the value of the group operation on the first element of the pair and g. This looks a bit strange in the raw set notation, when we write a function explicitely as a set of pairs and value of the group operation on the pair $\langle a, b\rangle$ as P⟨a,b⟩ instead of the usual infix $a \cdot b$ or $a + b$.

definition
 RightTranslation(G,P,g) ≡ –⟨ a,b⟩ ∈ G×G. P⟨a,g⟩ = b″

A similar definition of the left translation.

definition
 LeftTranslation(G,P,g) ≡ –⟨a,b⟩ ∈ G×G. P⟨g,a⟩ = b″

Translations map G into G. Two dimensional translations map $G \times G$ into itself.

lemma (in group0) group0˙5˙L1: assumes A1: g∈G
 shows RightTranslation(G,P,g) : G→G and LeftTranslation(G,P,g) : G→G
⟨*proof*⟩

The values of the translations are what we expect.

lemma (in group0) group0˙5˙L2: assumes g∈G a∈G
  shows
  RightTranslation(G,P,g)(a) = a·g
  LeftTranslation(G,P,g)(a) = g·a
  ⟨*proof*⟩

Composition of left translations is a left translation by the product.

lemma (in group0) group0˙5˙L4: assumes A1: g∈G h∈G a∈G and
  A2: T_g = LeftTranslation(G,P,g) T_h = LeftTranslation(G,P,h)
  shows
  $T_g(T_h(a))$ = g·h·a
  $T_g(T_h(a))$ = LeftTranslation(G,P,g·h)(a)
⟨*proof*⟩

Composition of right translations is a right translation by the product.

lemma (in group0) group0˙5˙L5: assumes A1: g∈G h∈G a∈G and
  A2: T_g = RightTranslation(G,P,g) T_h = RightTranslation(G,P,h)
  shows
 $T_g(T_h(a))$ = a·h·g
  $T_g(T_h(a))$ = RightTranslation(G,P,h·g)(a)
⟨*proof*⟩

Point free version of group0˙5˙L4 and group0˙5˙L5.

lemma (in group0) trans˙comp: assumes g∈G h∈G shows
 RightTranslation(G,P,g) O RightTranslation(G,P,h) = RightTranslation(G,P,h·g)
 LeftTranslation(G,P,g) O LeftTranslation(G,P,h) = LeftTranslation(G,P,g·h)
⟨*proof*⟩

The image of a set under a composition of translations is the same as the image under translation by a product.

lemma (in group0) trans˙comp˙image: assumes A1: g∈G h∈G and
  A2: T_g = LeftTranslation(G,P,g) T_h = LeftTranslation(G,P,h)
shows $T_g(T_h(A))$ = LeftTranslation(G,P,g·h)(A)
⟨*proof*⟩

Another form of the image of a set under a composition of translations

lemma (in group0) group0˙5˙L6:
  assumes A1: g∈G h∈G and A2: A⊆G and
  A3: T_g = RightTranslation(G,P,g) T_h = RightTranslation(G,P,h)
  shows $T_g(T_h(A))$ = –a·h·g. a∈A″
⟨*proof*⟩

The translation by neutral element is the identity on group.

lemma (in group0) trans˙neutral: shows
  RightTranslation(G,P,**1**) = id(G) and LeftTranslation(G,P,**1**) = id(G)
⟨*proof*⟩

Translation by neutral element does not move sets.

lemma (in group0) trans˙neutral˙image: assumes V⊆G
shows RightTranslation(G,P,**1**)(V) = V and LeftTranslation(G,P,**1**)(V) = V
⟨*proof*⟩

Composition of translations by an element and its inverse is identity.

lemma (in group0) trans˙comp˙id: assumes g∈G shows
RightTranslation(G,P,g) O RightTranslation(G,P,g^{-1}) = id(G) and
RightTranslation(G,P,g^{-1}) O RightTranslation(G,P,g) = id(G) and
LeftTranslation(G,P,g) O LeftTranslation(G,P,g^{-1}) = id(G) and
LeftTranslation(G,P,g^{-1}) O LeftTranslation(G,P,g) = id(G)
⟨*proof*⟩

Translations are bijective.

lemma (in group0) trans˙bij: assumes g∈G shows
RightTranslation(G,P,g) ∈ bij(G,G) and LeftTranslation(G,P,g) ∈ bij(G,G)
⟨*proof*⟩

Converse of a translation is translation by the inverse.

lemma (in group0) trans˙conv˙inv: assumes g∈G shows
converse(RightTranslation(G,P,g)) = RightTranslation(G,P,g^{-1}) and
converse(LeftTranslation(G,P,g)) = LeftTranslation(G,P,g^{-1}) and
LeftTranslation(G,P,g) = converse(LeftTranslation(G,P,g^{-1})) and
RightTranslation(G,P,g) = converse(RightTranslation(G,P,g^{-1}))
⟨*proof*⟩

The image of a set by translation is the same as the inverse image by by the inverse element translation.

lemma (in group0) trans˙image˙vimage: assumes g∈G shows
LeftTranslation(G,P,g)(A) = LeftTranslation(G,P,g^{-1})-(A) and
RightTranslation(G,P,g)(A) = RightTranslation(G,P,g^{-1})-(A)
⟨*proof*⟩

Another way of looking at translations is that they are sections of the group operation.

lemma (in group0) trans˙eq˙section: assumes g∈G shows
RightTranslation(G,P,g) = Fix2ndVar(P,g) and
LeftTranslation(G,P,g) = Fix1stVar(P,g)
⟨*proof*⟩

A lemma demonstrating what is the left translation of a set

lemma (in group0) ltrans˙image: assumes A1: V⊆G and A2: x∈G
shows LeftTranslation(G,P,x)(V) = –x·v. v∈V˝
⟨*proof*⟩

A lemma demonstrating what is the right translation of a set

lemma (in group0) rtrans˙image: assumes A1: V⊆G and A2: x∈G
shows RightTranslation(G,P,x)(V) = –v·x. v∈V˝

⟨*proof*⟩

Right and left translations of a set are subsets of the group. Interestingly, we do not have to assume the set is a subset of the group.

lemma (in group0) lrtrans˙in˙group: assumes x∈G
 shows LeftTranslation(G,P,x)(V) ⊆ G and RightTranslation(G,P,x)(V) ⊆ G
⟨*proof*⟩

A technical lemma about solving equations with translations.

lemma (in group0) ltrans˙inv˙in: assumes A1: V⊆G and A2: y∈G and
 A3: x ∈ LeftTranslation(G,P,y)(GroupInv(G,P)(V))
 shows y ∈ LeftTranslation(G,P,x)(V)
⟨*proof*⟩

We can look at the result of interval arithmetic operation as union of left translated sets.

lemma (in group0) image˙ltrans˙union: assumes A⊆G B⊆G shows
 (P –lifted to subsets of" G)⟨A,B⟩ = (⋃a∈A. LeftTranslation(G,P,a)(B))
⟨*proof*⟩

The right translation version of image˙ltrans˙union The proof follows the same schema.

lemma (in group0) image˙rtrans˙union: assumes A⊆G B⊆G shows
 (P –lifted to subsets of" G)⟨A,B⟩ = (⋃b∈B. RightTranslation(G,P,b)(A))
⟨*proof*⟩

If the neutral element belongs to a set, then an element of group belongs the translation of that set.

lemma (in group0) neut˙trans˙elem:
 assumes A1: A⊆G g∈G and A2: **1**∈A
 shows g ∈ LeftTranslation(G,P,g)(A) g ∈ RightTranslation(G,P,g)(A)
⟨*proof*⟩

The neutral element belongs to the translation of a set by the inverse of an element that belongs to it.

lemma (in group0) elem˙trans˙neut: assumes A1: A⊆G and A2: g∈A
 shows **1** ∈ LeftTranslation(G,P,g^{-1})(A) **1** ∈ RightTranslation(G,P,g^{-1})(A)
⟨*proof*⟩

27.2 Odd functions

This section is about odd functions.

Odd functions are those that commute with the group inverse: $f(a^{-1}) = (f(a))^{-1}$.

definition

IsOdd(G,P,f) ≡ (∀ a∈G. f(GroupInv(G,P)(a)) = GroupInv(G,P)(f(a)))

Let's see the definition of an odd function in a more readable notation.

lemma (in group0) group0˙6˙L1:
shows IsOdd(G,P,p) ⟷ (∀ a∈G. p(a^{-1}) = $(p(a))^{-1}$)
⟨*proof*⟩

We can express the definition of an odd function in two ways.

lemma (in group0) group0˙6˙L2:
assumes A1: p : G→G
shows
(∀ a∈G. p(a^{-1}) = $(p(a))^{-1}$) ⟷ (∀ a∈G. $(p(a^{-1}))^{-1}$ = p(a))
⟨*proof*⟩

27.3 Subgroups and interval arithmetic

The section Binary operations in the func˙ZF theory defines the notion of "lifting operation to subsets". In short, every binary operation $f : X \times X \longrightarrow X$ on a set X defines an operation on the subsets of X defined by $F(A,B) = \{f\langle x,y\rangle | x \in A, y \in B\}$. In the group context using multiplicative notation we can write this as $H \cdot K = \{x \cdot y | x \in A, y \in B\}$. Similarly we can define $H^{-1} = \{x^{-1} | x \in H\}$. In this section we study properties of these derived operation and how they relate to the concept of subgroups.

The next locale extends the groups0 locale with notation related to interval arithmetics.

locale group4 = group0 +
fixes sdot (infixl · 70)
defines sdot˙def [simp]: A·B ≡ (P –lifted to subsets of" G)⟨A,B⟩

fixes sinv (˙$^{-1}$ [90] 91)
defines sinv˙def[simp]: A^{-1} ≡ GroupInv(G,P)(A)

The next lemma shows a somewhat more explicit way of defining the product of two subsets of a group.

lemma (in group4) interval˙prod: assumes A⊆G B⊆G
shows A·B = –x·y. ⟨x,y⟩ ∈ A×B"
⟨*proof*⟩

Product of elements of subsets of the group is in the set product of those subsets

lemma (in group4) interval˙prod˙el: assumes A⊆G B⊆G x∈A y∈B
shows x·y ∈ A·B
⟨*proof*⟩

An alternative definition of a group inverse of a set.

lemma (in group4) interval˙inv: assumes A⊆G

shows $A^{-1} = \{x^{-1}.x \in A\}$
⟨*proof*⟩

Group inverse of a set is a subset of the group. Interestingly we don't need to assume the set is a subset of the group.

lemma (in group4) interval_inv_cl: shows $A^{-1} \subseteq G$
⟨*proof*⟩

The product of two subsets of a group is a subset of the group.

lemma (in group4) interval_prod_closed: assumes $A \subseteq G$ $B \subseteq G$
shows $A \cdot B \subseteq G$
⟨*proof*⟩

The product of sets operation is associative.

lemma (in group4) interval_prod_assoc: assumes $A \subseteq G$ $B \subseteq G$ $C \subseteq G$
shows $A \cdot B \cdot C = A \cdot (B \cdot C)$
⟨*proof*⟩

A simple rearrangement following from associativity of the product of sets operation.

lemma (in group4) interval_prod_rearr1: assumes $A \subseteq G$ $B \subseteq G$ $C \subseteq G$ $D \subseteq G$
shows $A \cdot B \cdot (C \cdot D) = A \cdot (B \cdot C) \cdot D$
⟨*proof*⟩

A subset A of the group is closed with respect to the group operation iff $A \cdot A \subseteq A$.

lemma (in group4) subset_gr_op_cl: assumes $A \subseteq G$
shows (A {is closed under} P) $\longleftrightarrow$ $A \cdot A \subseteq A$
⟨*proof*⟩

Inverse and square of a subgroup is this subgroup.

lemma (in group4) subgroup_inv_sq: assumes IsAsubgroup(H,P)
shows $H^{-1} = H$ and $H \cdot H = H$
⟨*proof*⟩

Inverse of a product two sets is a product of inverses with the reversed order.

lemma (in group4) interval_prod_inv: assumes $A \subseteq G$ $B \subseteq G$
shows
$(A \cdot B)^{-1} = \{(x \cdot y)^{-1}.\langle x,y \rangle \in A \times B\}$
$(A \cdot B)^{-1} = \{y^{-1} \cdot x^{-1}.\langle x,y \rangle \in A \times B\}$
$(A \cdot B)^{-1} = (B^{-1}) \cdot (A^{-1})$
⟨*proof*⟩

If H, K are subgroups then $H \cdot K$ is a subgroup iff $H \cdot K = K \cdot H$.

theorem (in group4) prod_subgr_subgr:
assumes IsAsubgroup(H,P) and IsAsubgroup(K,P)
shows IsAsubgroup(H·K,P) $\longleftrightarrow$ $H \cdot K = K \cdot H$

⟨*proof*⟩

end

28 Groups - and alternative definition

theory Group˙ZF˙1b imports Group˙ZF

begin

In a typical textbook a group is defined as a set G with an associative operation such that two conditions hold:

A: there is an element $e \in G$ such that for all $g \in G$ we have $e \cdot g = g$ and $g \cdot e = g$. We call this element a "unit" or a "neutral element" of the group.

B: for every $a \in G$ there exists a $b \in G$ such that $a \cdot b = e$, where e is the element of G whose existence is guaranteed by A.

The validity of this definition is rather dubious to me, as condition A does not define any specific element e that can be referred to in condition B - it merely states that a set of such units e is not empty. Of course it does work in the end as we can prove that the set of such neutral elements has exactly one element, but still the definition by itself is not valid. You just can't reference a variable bound by a quantifier outside of the scope of that quantifier.

One way around this is to first use condition A to define the notion of a monoid, then prove the uniqueness of e and then use the condition B to define groups.

Another way is to write conditions A and B together as follows:

$\exists_{e\in G} \ (\forall_{g\in G} \ e \cdot g = g \wedge g \cdot e = g) \wedge (\forall_{a\in G}\exists_{b\in G} \ a \cdot b = e)$.

This is rather ugly.

What I want to talk about is an amusing way to define groups directly without any reference to the neutral elements. Namely, we can define a group as a non-empty set G with an associative operation "$\cdot$" such that

C: for every $a, b \in G$ the equations $a \cdot x = b$ and $y \cdot a = b$ can be solved in G.

This theory file aims at proving the equivalence of this alternative definition with the usual definition of the group, as formulated in Group˙ZF.thy. The informal proofs come from an Aug. 14, 2005 post by buli on the matematyka.org forum.

28.1 An alternative definition of group

First we will define notation for writing about groups.

We will use the multiplicative notation for the group operation. To do this, we define a context (locale) that tells Isabelle to interpret $a \cdot b$ as the value of function P on the pair $\langle a, b\rangle$.

locale group2 =
  fixes P
  fixes dot (infixl · 70)
  defines dot˙def [simp]: a · b ≡ P⟨a,b⟩

The next theorem states that a set G with an associative operation that satisfies condition C is a group, as defined in IsarMathLib Group˙ZF theory.

theorem (in group2) altgroup˙is˙group:
  assumes A1: G≠0 and A2: P –is associative on″ G
  and A3: ∀a∈G.∀b∈G. ∃x∈G. a·x = b
  and A4: ∀a∈G.∀b∈G. ∃y∈G. y·a = b
  shows IsAgroup(G,P)
⟨*proof*⟩

The converse of altgroup˙is˙group: in every (classically defined) group condition C holds. In informal mathematics we can say "Obviously condition C holds in any group." In formalized mathematics the word "obviously" is not in the language. The next theorem is proven in the context called group0 defined in the theory Group˙ZF.thy. Similarly to the group2 that context defines $a \cdot b$ as $P\langle a, b\rangle$ It also defines notation related to the group inverse and adds an assumption that the pair (G, P) is a group to all its theorems. This is why in the next theorem we don't explicitely assume that (G, P) is a group - this assumption is implicit in the context.

theorem (in group0) group˙is˙altgroup: shows
  ∀a∈G.∀b∈G. ∃x∈G. a·x = b and ∀a∈G.∀b∈G. ∃y∈G. y·a = b
⟨*proof*⟩

end

29 Abelian Group

theory AbelianGroup˙ZF imports Group˙ZF

begin

A group is called "abelian" if its operation is commutative, i.e. $P\langle a, b\rangle = P\langle a, b\rangle$ for all group elements a, b, where P is the group operation. It is customary to use the additive notation for abelian groups, so this condition is typically written as $a+b = b+a$. We will be using multiplicative notation though (in which the commutativity condition of the operation is written as $a \cdot b = b \cdot a$), just to avoid the hassle of changing the notation we used for general groups.

29.1 Rearrangement formulae

This section is not interesting and should not be read. Here we will prove formulas is which right hand side uses the same factors as the left hand side, just in different order. These facts are obvious in informal math sense, but Isabelle prover is not able to derive them automatically, so we have to prove them by hand.

Proving the facts about associative and commutative operations is quite tedious in formalized mathematics. To a human the thing is simple: we can arrange the elements in any order and put parantheses wherever we want, it is all the same. However, formalizing this statement would be rather difficult (I think). The next lemma attempts a quasi-algorithmic approach to this type of problem. To prove that two expressions are equal, we first strip one from parantheses, then rearrange the elements in proper order, then put the parantheses where we want them to be. The algorithm for rearrangement is easy to describe: we keep putting the first element (from the right) that is in the wrong place at the left-most position until we get the proper arrangement. As far removing parantheses is concerned Isabelle does its job automatically.

lemma (in group0) group0˙4˙L2:
  assumes A1:P –is commutative on˝ G
  and A2:a∈G b∈G c∈G d∈G E∈G F∈G
  shows (a·b)·(c·d)·(E·F) = (a·(d·F))·(b·(c·E))
⟨*proof*⟩

Another useful rearrangement.

lemma (in group0) group0˙4˙L3:
  assumes A1:P –is commutative on˝ G
  and A2: a∈G b∈G and A3: c∈G d∈G E∈G F∈G
  shows a·b·((c·d)$^{-1}$·(E·F)$^{-1}$) = (a·(E·c)$^{-1}$)·(b·(F·d)$^{-1}$)
⟨*proof*⟩

Some useful rearrangements for two elements of a group.

lemma (in group0) group0˙4˙L4:
  assumes A1:P –is commutative on˝ G
  and A2: a∈G b∈G
  shows
  b^{-1}·a^{-1} = a^{-1}·b^{-1}
  (a·b)$^{-1}$ = a^{-1}·b^{-1}
  (a·b^{-1})$^{-1}$ = a^{-1}·b
⟨*proof*⟩

Another bunch of useful rearrangements with three elements.

lemma (in group0) group0˙4˙L4A:
  assumes A1: P –is commutative on˝ G
  and A2: a∈G b∈G c∈G

shows
$a \cdot b \cdot c = c \cdot a \cdot b$
$a^{-1} \cdot (b^{-1} \cdot c^{-1})^{-1} = (a \cdot (b \cdot c)^{-1})^{-1}$
$a \cdot (b \cdot c)^{-1} = a \cdot b^{-1} \cdot c^{-1}$
$a \cdot (b \cdot c^{-1})^{-1} = a \cdot b^{-1} \cdot c$
$a \cdot b^{-1} \cdot c^{-1} = a \cdot c^{-1} \cdot b^{-1}$
⟨*proof*⟩

Another useful rearrangement.

lemma (in group0) group0˙4˙L4B:
assumes P –is commutative on″ G
and a∈G b∈G c∈G
shows $a \cdot b^{-1} \cdot (b \cdot c^{-1}) = a \cdot c^{-1}$
⟨*proof*⟩

A couple of permutations of order for three alements.

lemma (in group0) group0˙4˙L4C:
assumes A1: P –is commutative on″ G
and A2: a∈G b∈G c∈G
shows
$a \cdot b \cdot c = c \cdot a \cdot b$
$a \cdot b \cdot c = a \cdot (c \cdot b)$
$a \cdot b \cdot c = c \cdot (a \cdot b)$
$a \cdot b \cdot c = c \cdot b \cdot a$
⟨*proof*⟩

Some rearangement with three elements and inverse.

lemma (in group0) group0˙4˙L4D:
assumes A1: P –is commutative on″ G
and A2: a∈G b∈G c∈G
shows
$a^{-1} \cdot b^{-1} \cdot c = c \cdot a^{-1} \cdot b^{-1}$
$b^{-1} \cdot a^{-1} \cdot c = c \cdot a^{-1} \cdot b^{-1}$
$(a^{-1} \cdot b \cdot c)^{-1} = a \cdot b^{-1} \cdot c^{-1}$
⟨*proof*⟩

Another rearrangement lemma with three elements and equation.

lemma (in group0) group0˙4˙L5: assumes A1:P –is commutative on″ G
and A2: a∈G b∈G c∈G
and A3: $c = a \cdot b^{-1}$
shows $a = b \cdot c$
⟨*proof*⟩

In abelian groups we can cancel an element with its inverse even if separated by another element.

lemma (in group0) group0˙4˙L6A: assumes A1: P –is commutative on″ G
and A2: a∈G b∈G
shows

$a \cdot b \cdot a^{-1} = b$
$a^{-1} \cdot b \cdot a = b$
$a^{-1} \cdot (b \cdot a) = b$
$a \cdot (b \cdot a^{-1}) = b$
⟨*proof*⟩

Another lemma about cancelling with two elements.

lemma (in group0) group0˙4˙L6AA:
assumes A1: P –is commutative on″ G and A2: $a \in G \quad b \in G$
shows $a \cdot b^{-1} \cdot a^{-1} = b^{-1}$
⟨*proof*⟩

Another lemma about cancelling with two elements.

lemma (in group0) group0˙4˙L6AB:
assumes A1: P –is commutative on″ G and A2: $a \in G \quad b \in G$
shows
$a \cdot (a \cdot b)^{-1} = b^{-1}$
$a \cdot (b \cdot a^{-1}) = b$
⟨*proof*⟩

Another lemma about cancelling with two elements.

lemma (in group0) group0˙4˙L6AC:
assumes P –is commutative on″ G and $a \in G \quad b \in G$
shows $a \cdot (a \cdot b^{-1})^{-1} = b$
⟨*proof*⟩

In abelian groups we can cancel an element with its inverse even if separated by two other elements.

lemma (in group0) group0˙4˙L6B: assumes A1: P –is commutative on″ G
and A2: $a \in G \quad b \in G \quad c \in G$
shows
$a \cdot b \cdot c \cdot a^{-1} = b \cdot c$
$a^{-1} \cdot b \cdot c \cdot a = b \cdot c$
⟨*proof*⟩

In abelian groups we can cancel an element with its inverse even if separated by three other elements.

lemma (in group0) group0˙4˙L6C: assumes A1: P –is commutative on″ G
and A2: $a \in G$ $b \in G$ $c \in G$ $d \in G$
shows $a \cdot b \cdot c \cdot d \cdot a^{-1} = b \cdot c \cdot d$
⟨*proof*⟩

Another couple of useful rearrangements of three elements and cancelling.

lemma (in group0) group0˙4˙L6D:
assumes A1: P –is commutative on″ G
and A2: $a \in G \quad b \in G \quad c \in G$
shows

$a \cdot b^{-1} \cdot (a \cdot c^{-1})^{-1} = c \cdot b^{-1}$
$(a \cdot c)^{-1} \cdot (b \cdot c) = a^{-1} \cdot b$
$a \cdot (b \cdot (c \cdot a^{-1} \cdot b^{-1})) = c$
$a \cdot b \cdot c^{-1} \cdot (c \cdot a^{-1}) = b$
⟨*proof*⟩

Another useful rearrangement of three elements and cancelling.

lemma (in group0) group0˙4˙L6E:
assumes A1: P –is commutative on″ G
and A2: a∈G b∈G c∈G
shows
$a \cdot b \cdot (a \cdot c)^{-1} = b \cdot c^{-1}$
⟨*proof*⟩

A rearrangement with two elements and canceelling, special case of group0˙4˙L6D when $c = b^{-1}$.

lemma (in group0) group0˙4˙L6F:
assumes A1: P –is commutative on″ G
and A2: a∈G b∈G
shows $a \cdot b^{-1} \cdot (a \cdot b)^{-1} = b^{-1} \cdot b^{-1}$
⟨*proof*⟩

Some other rearrangements with four elements. The algorithm for proof as in group0˙4˙L2 works very well here.

lemma (in group0) rearr˙ab˙gr˙4˙elemA:
assumes A1: P –is commutative on″ G
and A2: a∈G b∈G c∈G d∈G
shows
$a \cdot b \cdot c \cdot d = a \cdot d \cdot b \cdot c$
$a \cdot b \cdot c \cdot d = a \cdot c \cdot (b \cdot d)$
⟨*proof*⟩

Some rearrangements with four elements and inverse that are applications of rearr˙ab˙gr˙4˙elem

lemma (in group0) rearr˙ab˙gr˙4˙elemB:
assumes A1: P –is commutative on″ G
and A2: a∈G b∈G c∈G d∈G
shows
$a \cdot b^{-1} \cdot c^{-1} \cdot d^{-1} = a \cdot d^{-1} \cdot b^{-1} \cdot c^{-1}$
$a \cdot b \cdot c \cdot d^{-1} = a \cdot d^{-1} \cdot b \cdot c$
$a \cdot b \cdot c^{-1} \cdot d^{-1} = a \cdot c^{-1} \cdot (b \cdot d^{-1})$
⟨*proof*⟩

Some rearrangement lemmas with four elements.

lemma (in group0) group0˙4˙L7:
assumes A1: P –is commutative on″ G
and A2: a∈G b∈G c∈G d∈G
shows

$a \cdot b \cdot c \cdot d^{-1} = a \cdot d^{-1} \cdot b \cdot c$
$a \cdot d \cdot (b \cdot d \cdot (c \cdot d))^{-1} = a \cdot (b \cdot c)^{-1} \cdot d^{-1}$
$a \cdot (b \cdot c) \cdot d = a \cdot b \cdot d \cdot c$
⟨*proof*⟩

Some other rearrangements with four elements.

lemma (in group0) group0˙4˙L8:
assumes A1: P –is commutative on″ G
and A2: a∈G b∈G c∈G d∈G
shows
$a \cdot (b \cdot c)^{-1} = (a \cdot d^{-1} \cdot c^{-1}) \cdot (d \cdot b^{-1})$
$a \cdot b \cdot (c \cdot d) = c \cdot a \cdot (b \cdot d)$
$a \cdot b \cdot (c \cdot d) = a \cdot c \cdot (b \cdot d)$
$a \cdot (b \cdot c^{-1}) \cdot d = a \cdot b \cdot d \cdot c^{-1}$
$(a \cdot b) \cdot (c \cdot d)^{-1} \cdot (b \cdot d^{-1})^{-1} = a \cdot c^{-1}$
⟨*proof*⟩

Some other rearrangements with four elements.

lemma (in group0) group0˙4˙L8A:
assumes A1: P –is commutative on″ G
and A2: a∈G b∈G c∈G d∈G
shows
$a \cdot b^{-1} \cdot (c \cdot d^{-1}) = a \cdot c \cdot (b^{-1} \cdot d^{-1})$
$a \cdot b^{-1} \cdot (c \cdot d^{-1}) = a \cdot c \cdot b^{-1} \cdot d^{-1}$
⟨*proof*⟩

Some rearrangements with an equation.

lemma (in group0) group0˙4˙L9:
assumes A1: P –is commutative on″ G
and A2: a∈G b∈G c∈G d∈G
and A3: $a = b \cdot c^{-1} \cdot d^{-1}$
shows
$d = b \cdot a^{-1} \cdot c^{-1}$
$d = a^{-1} \cdot b \cdot c^{-1}$
$b = a \cdot d \cdot c$
⟨*proof*⟩

end

30 Groups 2

theory Group˙ZF˙2 imports AbelianGroup˙ZF func˙ZF EquivClass1

begin

This theory continues Group_ZF.thy and considers lifting the group structure to function spaces and projecting the group structure to quotient spaces, in particular the quotient qroup.

30.1 Lifting groups to function spaces

If we have a monoid (group) G than we get a monoid (group) structure on a space of functions valued in in G by defining $(f \cdot g)(x) := f(x) \cdot g(x)$. We call this process "lifting the monoid (group) to function space". This section formalizes this lifting.

The lifted operation is an operation on the function space.

lemma (in monoid0) Group˙ZF˙2˙1˙L0A:
  assumes A1: F = f –lifted to function space over˝ X
  shows F : (X→G)×(X→G)→(X→G)
⟨*proof*⟩

The result of the lifted operation is in the function space.

lemma (in monoid0) Group˙ZF˙2˙1˙L0:
  assumes A1:F = f –lifted to function space over˝ X
  and A2:s:X→G r:X→G
  shows F⟨ s,r⟩ : X→G
⟨*proof*⟩

The lifted monoid operation has a neutral element, namely the constant function with the neutral element as the value.

lemma (in monoid0) Group˙ZF˙2˙1˙L1:
  assumes A1: F = f –lifted to function space over˝ X
  and A2: E = ConstantFunction(X,TheNeutralElement(G,f))
  shows E : X→G ∧ (∀s∈X→G. F⟨ E,s⟩ = s ∧ F⟨ s,E⟩ = s)
⟨*proof*⟩

Monoids can be lifted to a function space.

lemma (in monoid0) Group˙ZF˙2˙1˙T1:
  assumes A1: F = f –lifted to function space over˝ X
  shows IsAmonoid(X→G,F)
⟨*proof*⟩

The constant function with the neutral element as the value is the neutral element of the lifted monoid.

lemma Group˙ZF˙2˙1˙L2:
  assumes A1: IsAmonoid(G,f)
  and A2: F = f –lifted to function space over˝ X
  and A3: E = ConstantFunction(X,TheNeutralElement(G,f))
  shows E = TheNeutralElement(X→G,F)
⟨*proof*⟩

The lifted operation acts on the functions in a natural way defined by the monoid operation.

lemma (in monoid0) lifted˙val:
  assumes F = f –lifted to function space over˝ X

and s:X→G r:X→G
and x∈X
shows (F⟨s,r⟩)(x) = s(x) ⊕ r(x)
⟨*proof*⟩

The lifted operation acts on the functions in a natural way defined by the group operation. This is the same as lifted˙val, but in the group0 context.

lemma (in group0) Group˙ZF˙2˙1˙L3:
assumes F = P –lifted to function space over″ X
and s:X→G r:X→G
and x∈X
shows (F⟨s,r⟩)(x) = s(x)·r(x)
⟨*proof*⟩

In the group0 context we can apply theorems proven in monoid0 context to the lifted monoid.

lemma (in group0) Group˙ZF˙2˙1˙L4:
assumes A1: F = P –lifted to function space over″ X
shows monoid0(X→G,F)
⟨*proof*⟩

The compostion of a function $f : X \to G$ with the group inverse is a right inverse for the lifted group.

lemma (in group0) Group˙ZF˙2˙1˙L5:
assumes A1: F = P –lifted to function space over″ X
and A2: s : X→G
and A3: i = GroupInv(G,P) O s
shows i: X→G and F⟨ s,i⟩ = TheNeutralElement(X→G,F)
⟨*proof*⟩

Groups can be lifted to the function space.

theorem (in group0) Group˙ZF˙2˙1˙T2:
assumes A1: F = P –lifted to function space over″ X
shows IsAgroup(X→G,F)
⟨*proof*⟩

What is the group inverse for the lifted group?

lemma (in group0) Group˙ZF˙2˙1˙L6:
assumes A1: F = P –lifted to function space over″ X
shows ∀s∈(X→G). GroupInv(X→G,F)(s) = GroupInv(G,P) O s
⟨*proof*⟩

What is the value of the group inverse for the lifted group?

corollary (in group0) lift˙gr˙inv˙val:
assumes F = P –lifted to function space over″ X and
s : X→G and x∈X
shows (GroupInv(X→G,F)(s))(x) = $(s(x))^{-1}$

⟨*proof*⟩

What is the group inverse in a subgroup of the lifted group?

lemma (in group0) Group˙ZF˙2˙1˙L6A:
  assumes A1: F = P –lifted to function space over″ X
  and A2: IsAsubgroup(H,F)
  and A3: g = restrict(F,H×H)
  and A4: s∈H
  shows GroupInv(H,g)(s) = GroupInv(G,P) O s
⟨*proof*⟩

If a group is abelian, then its lift to a function space is also abelian.

lemma (in group0) Group˙ZF˙2˙1˙L7:
  assumes A1: F = P –lifted to function space over″ X
  and A2: P –is commutative on″ G
  shows F –is commutative on″ (X→G)
⟨*proof*⟩

30.2 Equivalence relations on groups

The goal of this section is to establish that (under some conditions) given an equivalence relation on a group or (monoid)we can project the group (monoid) structure on the quotient and obtain another group.

The neutral element class is neutral in the projection.

lemma (in monoid0) Group˙ZF˙2˙2˙L1:
  assumes A1: equiv(G,r) and A2:Congruent2(r,f)
  and A3: F = ProjFun2(G,r,f)
  and A4: e = TheNeutralElement(G,f)
  shows r–e″ ∈ G//r ∧
  (∀c ∈ G//r. F⟨ r–e″,c⟩ = c ∧ F⟨ c,r–e″⟩ = c)
⟨*proof*⟩

The projected structure is a monoid.

theorem (in monoid0) Group˙ZF˙2˙2˙T1:
  assumes A1: equiv(G,r) and A2: Congruent2(r,f)
  and A3: F = ProjFun2(G,r,f)
  shows IsAmonoid(G//r,F)
⟨*proof*⟩

The class of the neutral element is the neutral element of the projected monoid.

lemma Group˙ZF˙2˙2˙L1:
  assumes A1: IsAmonoid(G,f)
  and A2: equiv(G,r) and A3: Congruent2(r,f)
  and A4: F = ProjFun2(G,r,f)
  and A5: e = TheNeutralElement(G,f)

shows r–e″ = TheNeutralElement(G//r,F)
⟨*proof*⟩

The projected operation can be defined in terms of the group operation on representants in a natural way.

lemma (in group0) Group˙ZF˙2˙2˙L2:
assumes A1: equiv(G,r) and A2: Congruent2(r,P)
and A3: F = ProjFun2(G,r,P)
and A4: a∈G b∈G
shows F⟨ r–a″,r–b″⟩ = r–a·b″
⟨*proof*⟩

The class of the inverse is a right inverse of the class.

lemma (in group0) Group˙ZF˙2˙2˙L3:
assumes A1: equiv(G,r) and A2: Congruent2(r,P)
and A3: F = ProjFun2(G,r,P)
and A4: a∈G
shows F⟨r–a″,r–a^{-1}″⟩ = TheNeutralElement(G//r,F)
⟨*proof*⟩

The group structure can be projected to the quotient space.

theorem (in group0) Group˙ZF˙3˙T2:
assumes A1: equiv(G,r) and A2: Congruent2(r,P)
shows IsAgroup(G//r,ProjFun2(G,r,P))
⟨*proof*⟩

The group inverse (in the projected group) of a class is the class of the inverse.

lemma (in group0) Group˙ZF˙2˙2˙L4:
assumes A1: equiv(G,r) and
A2: Congruent2(r,P) and
A3: F = ProjFun2(G,r,P) and
A4: a∈G
shows r–a^{-1}″ = GroupInv(G//r,F)(r–a″)
⟨*proof*⟩

30.3 Normal subgroups and quotient groups

If H is a subgroup of G, then for every $a \in G$ we can cosider the sets $\{a \cdot h.h \in H\}$ and $\{h \cdot a.h \in H\}$ (called a left and right "coset of H", resp.) These sets sometimes form a group, called the "quotient group". This section discusses the notion of quotient groups.

A normal subgorup N of a group G is such that aba^{-1} belongs to N if $a \in G, b \in N$.

definition
IsAnormalSubgroup(G,P,N) ≡ IsAsubgroup(N,P) ∧

(∀n∈N.∀g∈G. P⟨ P⟨ g,n ⟩,GroupInv(G,P)(g) ⟩ ∈ N)

Having a group and a normal subgroup N we can create another group consisting of eqivalence classes of the relation $a \sim b \equiv a \cdot b^{-1} \in N$. We will refer to this relation as the quotient group relation. The classes of this relation are in fact cosets of subgroup H.

definition
QuotientGroupRel(G,P,H) ≡
–⟨ a,b⟩ ∈ G×G. P⟨ a, GroupInv(G,P)(b)⟩ ∈ H″

Next we define the operation in the quotient group as the projection of the group operation on the classses of the quotient group relation.

definition
QuotientGroupOp(G,P,H) ≡ ProjFun2(G,QuotientGroupRel(G,P,H),P)

Definition of a normal subgroup in a more readable notation.

lemma (in group0) Group˙ZF˙2˙4˙L0:
assumes IsAnormalSubgroup(G,P,H)
and g∈G n∈H
shows g·n·g^{-1} ∈ H
⟨*proof*⟩

The quotient group relation is reflexive.

lemma (in group0) Group˙ZF˙2˙4˙L1:
assumes IsAsubgroup(H,P)
shows refl(G,QuotientGroupRel(G,P,H))
⟨*proof*⟩

The quotient group relation is symmetric.

lemma (in group0) Group˙ZF˙2˙4˙L2:
assumes A1:IsAsubgroup(H,P)
shows sym(QuotientGroupRel(G,P,H))
⟨*proof*⟩

The quotient group relation is transistive.

lemma (in group0) Group˙ZF˙2˙4˙L3A:
assumes A1: IsAsubgroup(H,P) and
A2: ⟨ a,b⟩ ∈ QuotientGroupRel(G,P,H) and
A3: ⟨ b,c⟩ ∈ QuotientGroupRel(G,P,H)
shows ⟨ a,c⟩ ∈ QuotientGroupRel(G,P,H)
⟨*proof*⟩

The quotient group relation is an equivalence relation. Note we do not need the subgroup to be normal for this to be true.

lemma (in group0) Group˙ZF˙2˙4˙L3: assumes A1:IsAsubgroup(H,P)
shows equiv(G,QuotientGroupRel(G,P,H))
⟨*proof*⟩

The next lemma states the essential condition for congruency of the group operation with respect to the quotient group relation.

lemma (in group0) Group˙ZF˙2˙4˙L4:
  assumes A1: IsAnormalSubgroup(G,P,H)
  and A2: ⟨a1,a2⟩ ∈ QuotientGroupRel(G,P,H)
  and A3: ⟨b1,b2⟩ ∈ QuotientGroupRel(G,P,H)
  shows ⟨a1·b1, a2·b2⟩ ∈ QuotientGroupRel(G,P,H)
⟨*proof*⟩

If the subgroup is normal, the group operation is congruent with respect to the quotient group relation.

lemma Group˙ZF˙2˙4˙L5A:
  assumes IsAgroup(G,P)
  and IsAnormalSubgroup(G,P,H)
  shows Congruent2(QuotientGroupRel(G,P,H),P)
  ⟨*proof*⟩

The quotient group is indeed a group.

theorem Group˙ZF˙2˙4˙T1:
  assumes IsAgroup(G,P) and IsAnormalSubgroup(G,P,H)
  shows
  IsAgroup(G//QuotientGroupRel(G,P,H),QuotientGroupOp(G,P,H))
  ⟨*proof*⟩

The class (coset) of the neutral element is the neutral element of the quotient group.

lemma Group˙ZF˙2˙4˙L5B:
  assumes IsAgroup(G,P) and IsAnormalSubgroup(G,P,H)
  and r = QuotientGroupRel(G,P,H)
  and e = TheNeutralElement(G,P)
  shows r–e″ = TheNeutralElement(G//r,QuotientGroupOp(G,P,H))
  ⟨*proof*⟩

A group element is equivalent to the neutral element iff it is in the subgroup we divide the group by.

lemma (in group0) Group˙ZF˙2˙4˙L5C: assumes a∈G
  shows ⟨a,**1**⟩ ∈ QuotientGroupRel(G,P,H) ⟷ a∈H
  ⟨*proof*⟩

A group element is in H iff its class is the neutral element of G/H.

lemma (in group0) Group˙ZF˙2˙4˙L5D:
  assumes A1: IsAnormalSubgroup(G,P,H) and
  A2: a∈G and
  A3: r = QuotientGroupRel(G,P,H) and
  A4: TheNeutralElement(G//r,QuotientGroupOp(G,P,H)) = e
  shows r–a″ = e ⟷ ⟨a,**1**⟩ ∈ r
⟨*proof*⟩

The class of $a \in G$ is the neutral element of the quotient G/H iff $a \in H$.

lemma (in group0) Group˙ZF˙2˙4˙L5E:
  assumes IsAnormalSubgroup(G,P,H) and
  a∈G and r = QuotientGroupRel(G,P,H) and
  TheNeutralElement(G//r,QuotientGroupOp(G,P,H)) = e
  shows r–a″ = e ⟷ a∈H
  ⟨*proof*⟩

Essential condition to show that every subgroup of an abelian group is normal.

lemma (in group0) Group˙ZF˙2˙4˙L5:
  assumes A1: P –is commutative on″ G
  and A2: IsAsubgroup(H,P)
  and A3: g∈G h∈H
  shows g·h·g^{-1} ∈ H
⟨*proof*⟩

Every subgroup of an abelian group is normal. Moreover, the quotient group is also abelian.

lemma Group˙ZF˙2˙4˙L6:
  assumes A1: IsAgroup(G,P)
  and A2: P –is commutative on″ G
  and A3: IsAsubgroup(H,P)
  shows IsAnormalSubgroup(G,P,H)
  QuotientGroupOp(G,P,H) –is commutative on″ (G//QuotientGroupRel(G,P,H))
⟨*proof*⟩

The group inverse (in the quotient group) of a class (coset) is the class of the inverse.

lemma (in group0) Group˙ZF˙2˙4˙L7:
  assumes IsAnormalSubgroup(G,P,H)
  and a∈G and r = QuotientGroupRel(G,P,H)
  and F = QuotientGroupOp(G,P,H)
  shows r–a^{-1}″ = GroupInv(G//r,F)(r–a″)
  ⟨*proof*⟩

30.4 Function spaces as monoids

On every space of functions $\{f : X \to X\}$ we can define a natural monoid structure with composition as the operation. This section explores this fact.

The next lemma states that composition has a neutral element, namely the identity function on X (the one that maps $x \in X$ into itself).

lemma Group˙ZF˙2˙5˙L1: assumes A1: F = Composition(X)
  shows ∃I∈(X→X). ∀f∈(X→X). F⟨ I,f⟩ = f ∧ F⟨ f,I⟩ = f
⟨*proof*⟩

The space of functions that map a set X into itsef is a monoid with composition as operation and the identity function as the neutral element.

lemma Group˙ZF˙2˙5˙L2: shows
 IsAmonoid(X→X,Composition(X))
 id(X) = TheNeutralElement(X→X,Composition(X))
⟨*proof*⟩

end

31 Groups 3

theory Group˙ZF˙3 imports Group˙ZF˙2 Finite1

begin

In this theory we consider notions in group theory that are useful for the construction of real numbers in the Real˙ZF˙x series of theories.

31.1 Group valued finite range functions

In this section show that the group valued functions $f : X \to G$, with the property that $f(X)$ is a finite subset of G, is a group. Such functions play an important role in the construction of real numbers in the Real˙ZF series.

The following proves the essential condition to show that the set of finite range functions is closed with respect to the lifted group operation.

lemma (in group0) Group˙ZF˙3˙1˙L1:
 assumes A1: F = P –lifted to function space over˝ X
 and
 A2: s ∈ FinRangeFunctions(X,G) r ∈ FinRangeFunctions(X,G)
 shows F⟨ s,r⟩ ∈ FinRangeFunctions(X,G)
⟨*proof*⟩

The set of group valued finite range functions is closed with respect to the lifted group operation.

lemma (in group0) Group˙ZF˙3˙1˙L2:
 assumes A1: F = P –lifted to function space over˝ X
 shows FinRangeFunctions(X,G) –is closed under˝ F
⟨*proof*⟩

A composition of a finite range function with the group inverse is a finite range function.

lemma (in group0) Group˙ZF˙3˙1˙L3:
 assumes A1: s ∈ FinRangeFunctions(X,G)
 shows GroupInv(G,P) O s ∈ FinRangeFunctions(X,G)
 ⟨*proof*⟩

The set of finite range functions is s subgroup of the lifted group.

theorem Group˙ZF˙3˙1˙T1:
  assumes A1: IsAgroup(G,P)
  and A2: F = P –lifted to function space over˝ X
  and A3: X≠0
  shows IsAsubgroup(FinRangeFunctions(X,G),F)
⟨*proof*⟩

31.2 Almost homomorphisms

An almost homomorphism is a group valued function defined on a monoid M with the property that the set $\{f(m+n)-f(m)-f(n)\}_{m,n\in M}$ is finite. This term is used by R. D. Arthan in "The Eudoxus Real Numbers". We use this term in the general group context and use the A'Campo's term "slopes" (see his "A natural construction for the real numbers") to mean an almost homomorphism mapping interegers into themselves. We consider almost homomorphisms because we use slopes to define real numbers in the Real˙ZF˙x series.

HomDiff is an acronym for "homomorphism difference". This is the expression $s(mn)(s(m)s(n))^{-1}$, or $s(m+n)-s(m)-s(n)$ in the additive notation. It is equal to the neutral element of the group if s is a homomorphism.

definition
  HomDiff(G,f,s,x) ≡
  f⟨s(f⟨ fst(x),snd(x)⟩) ,
  (GroupInv(G,f)(f⟨ s(fst(x)),s(snd(x))⟩))⟩

Almost homomorphisms are defined as those maps $s : G \to G$ such that the homomorphism difference takes only finite number of values on $G \times G$.

definition
  AlmostHoms(G,f) ≡
  –s ∈ G→G.–HomDiff(G,f,s,x). x ∈ G×G ˝ ∈ Fin(G)˝

AlHomOp1(G,f) is the group operation on almost homomorphisms defined in a natural way by $(s\cdot r)(n) = s(n)\cdot r(n)$. In the terminology defined in func1.thy this is the group operation f (on G) lifted to the function space $G \to G$ and restricted to the set AlmostHoms(G,f).

definition
  AlHomOp1(G,f) ≡
  restrict(f –lifted to function space over˝ G,
  AlmostHoms(G,f)×AlmostHoms(G,f))

We also define a composition (binary) operator on almost homomorphisms in a natural way. We call that operator AlHomOp2 - the second operation on almost homomorphisms. Composition of almost homomorphisms is used to define multiplication of real numbers in Real˙ZF series.

definition
AlHomOp2(G,f) ≡
restrict(Composition(G),AlmostHoms(G,f)×AlmostHoms(G,f))

This lemma provides more readable notation for the HomDiff definition. Not really intended to be used in proofs, but just to see the definition in the notation defined in the group0 locale.

lemma (in group0) HomDiff˙notation:
shows HomDiff(G,P,s,⟨ m,n⟩) = s(m·n)·(s(m)·s(n))$^{-1}$
⟨*proof*⟩

The next lemma shows the set from the definition of almost homomorphism in a different form.

lemma (in group0) Group˙ZF˙3˙2˙L1A: shows
–HomDiff(G,P,s,x). x ∈ G×G ″ = –s(m·n)·(s(m)·s(n))$^{-1}$. ⟨ m,n⟩ ∈ G×G″
⟨*proof*⟩

Let's define some notation. We inherit the notation and assumptions from the group0 context (locale) and add some. We will use AH to denote the set of almost homomorphisms. $\sim$ is the inverse (negative if the group is the group of integers) of almost homomorphisms, $(\sim p)(n) = p(n)^{-1}$. δ will denote the homomorphism difference specific for the group (HomDiff(G, f)). The notation $s \approx r$ will mean that s, r are almost equal, that is they are in the equivalence relation defined by the group of finite range functions (that is a normal subgroup of almost homomorphisms, if the group is abelian). We show that this is equivalent to the set $\{s(n) \cdot r(n)^{-1} : n \in G\}$ being finite. We also add an assumption that the G is abelian as many needed properties do not hold without that.

locale group1 = group0 +
assumes isAbelian: P –is commutative on″ G

fixes AH
defines AH˙def [simp]: AH ≡ AlmostHoms(G,P)

fixes Op1
defines Op1˙def [simp]: Op1 ≡ AlHomOp1(G,P)

fixes Op2
defines Op2˙def [simp]: Op2 ≡ AlHomOp2(G,P)

fixes FR
defines FR˙def [simp]: FR ≡ FinRangeFunctions(G,G)

fixes neg (∼˙ [90] 91)
defines neg˙def [simp]: ∼s ≡ GroupInv(G,P) O s

fixes δ

defines δ˙def [simp]: δ(s,x) ≡ HomDiff(G,P,s,x)

fixes AHprod (infix · 69)
defines AHprod˙def [simp]: s · r ≡ AlHomOp1(G,P)⟨s,r⟩

fixes AHcomp (infix ∘ 70)
defines AHcomp˙def [simp]: s ∘ r ≡ AlHomOp2(G,P)⟨s,r⟩

fixes AlEq (infix ≈ 68)
defines AlEq˙def [simp]:
s ≈ r ≡ ⟨s,r⟩ ∈ QuotientGroupRel(AH,Op1,FR)

HomDiff is a homomorphism on the lifted group structure.

lemma (in group1) Group˙ZF˙3˙2˙L1:
assumes A1: s:G→G r:G→G
and A2: x ∈ G×G
and A3: F = P –lifted to function space over″ G
shows δ(F⟨ s,r⟩,x) = δ(s,x)·δ(r,x)
⟨*proof*⟩

The group operation lifted to the function space over G preserves almost homomorphisms.

lemma (in group1) Group˙ZF˙3˙2˙L2: assumes A1: s ∈ AH r ∈ AH
and A2: F = P –lifted to function space over″ G
shows F⟨ s,r⟩ ∈ AH
⟨*proof*⟩

The set of almost homomorphisms is closed under the lifted group operation.

lemma (in group1) Group˙ZF˙3˙2˙L3:
assumes F = P –lifted to function space over″ G
shows AH –is closed under″ F
⟨*proof*⟩

The terms in the homomorphism difference for a function are in the group.

lemma (in group1) Group˙ZF˙3˙2˙L4:
assumes s:G→G and m∈G n∈G
shows
m·n ∈ G
s(m·n) ∈ G
s(m) ∈ G s(n) ∈ G
δ(s,⟨ m,n⟩) ∈ G
s(m)·s(n) ∈ G
⟨*proof*⟩

It is handy to have a version of Group˙ZF˙3˙2˙L4 specifically for almost homomorphisms.

corollary (in group1) Group˙ZF˙3˙2˙L4A:
assumes s ∈ AH and m∈G n∈G

shows $m \cdot n \in G$
$s(m \cdot n) \in G$
$s(m) \in G$ $s(n) \in G$
$\delta(s,\langle m,n\rangle) \in G$
$s(m) \cdot s(n) \in G$
⟨*proof*⟩

The terms in the homomorphism difference are in the group, a different form.

lemma (in group1) Group˙ZF˙3˙2˙L4B:
assumes A1:$s \in AH$ and A2:$x \in G \times G$
shows $fst(x) \cdot snd(x) \in G$
$s(fst(x) \cdot snd(x)) \in G$
$s(fst(x)) \in G$ $s(snd(x)) \in G$
$\delta(s,x) \in G$
$s(fst(x)) \cdot s(snd(x)) \in G$
⟨*proof*⟩

What are the values of the inverse of an almost homomorphism?

lemma (in group1) Group˙ZF˙3˙2˙L5:
assumes $s \in AH$ and $n \in G$
shows $(\sim s)(n) = (s(n))^{-1}$
⟨*proof*⟩

Homomorphism difference commutes with the inverse for almost homomorphisms.

lemma (in group1) Group˙ZF˙3˙2˙L6:
assumes A1:$s \in AH$ and A2:$x \in G \times G$
shows $\delta(\sim s,x) = (\delta(s,x))^{-1}$
⟨*proof*⟩

The inverse of an almost homomorphism maps the group into itself.

lemma (in group1) Group˙ZF˙3˙2˙L7:
assumes $s \in AH$
shows $\sim s : G \to G$
⟨*proof*⟩

The inverse of an almost homomorphism is an almost homomorphism.

lemma (in group1) Group˙ZF˙3˙2˙L8:
assumes A1: F = P –lifted to function space over″ G
and A2: $s \in AH$
shows GroupInv$(G \to G,F)(s) \in AH$
⟨*proof*⟩

The function that assigns the neutral element everywhere is an almost homomorphism.

lemma (in group1) Group˙ZF˙3˙2˙L9: shows

ConstantFunction(G,**1**) ∈ AH and AH≠0
⟨*proof*⟩

If the group is abelian, then almost homomorphisms form a subgroup of the lifted group.

lemma Group˙ZF˙3˙2˙L10:
  assumes A1: IsAgroup(G,P)
  and A2: P –is commutative on˝ G
  and A3: F = P –lifted to function space over˝ G
  shows IsAsubgroup(AlmostHoms(G,P),F)
⟨*proof*⟩

If the group is abelian, then almost homomorphisms form a group with the first operation, hence we can use theorems proven in group0 context aplied to this group.

lemma (in group1) Group˙ZF˙3˙2˙L10A:
  shows IsAgroup(AH,Op1) group0(AH,Op1)
    ⟨*proof*⟩

The group of almost homomorphisms is abelian

lemma Group˙ZF˙3˙2˙L11: assumes A1: IsAgroup(G,f)
  and A2: f –is commutative on˝ G
  shows
  IsAgroup(AlmostHoms(G,f),AlHomOp1(G,f))
  AlHomOp1(G,f) –is commutative on˝ AlmostHoms(G,f)
⟨*proof*⟩

The first operation on homomorphisms acts in a natural way on its operands.

lemma (in group1) Group˙ZF˙3˙2˙L12:
  assumes s∈AH r∈AH and n∈G
  shows (s•r)(n) = s(n)·r(n)
  ⟨*proof*⟩

What is the group inverse in the group of almost homomorphisms?

lemma (in group1) Group˙ZF˙3˙2˙L13:
  assumes A1: s∈AH
  shows
  GroupInv(AH,Op1)(s) = GroupInv(G,P) O s
  GroupInv(AH,Op1)(s) ∈ AH
  GroupInv(G,P) O s ∈ AH
⟨*proof*⟩

The group inverse in the group of almost homomorphisms acts in a natural way on its operand.

lemma (in group1) Group˙ZF˙3˙2˙L14:
  assumes s∈AH and n∈G
  shows (GroupInv(AH,Op1)(s))(n) = $(s(n))^{-1}$

⟨*proof*⟩

The next lemma states that if s,r are almost homomorphisms, then $s \cdot r^{-1}$ is also an almost homomorphism.

lemma Group˙ZF˙3˙2˙L15: assumes IsAgroup(G,f)
and f –is commutative on″ G
and AH = AlmostHoms(G,f) Op1 = AlHomOp1(G,f)
and s ∈ AH r ∈ AH
shows
Op1⟨ s,r⟩ ∈ AH
GroupInv(AH,Op1)(r) ∈ AH
Op1⟨ s,GroupInv(AH,Op1)(r)⟩ ∈ AH
⟨*proof*⟩

A version of Group˙ZF˙3˙2˙L15 formulated in notation used in group1 context. States that the product of almost homomorphisms is an almost homomorphism and the the product of an almost homomorphism with a (pointwise) inverse of an almost homomorphism is an almost homomorphism.

corollary (in group1) Group˙ZF˙3˙2˙L16: assumes s ∈ AH r ∈ AH
shows s·r ∈ AH s·(∼r) ∈ AH
⟨*proof*⟩

31.3 The classes of almost homomorphisms

In the Real˙ZF series we define real numbers as a quotient of the group of integer almost homomorphisms by the integer finite range functions. In this section we setup the background for that in the general group context.

Finite range functions are almost homomorphisms.

lemma (in group1) Group˙ZF˙3˙3˙L1: shows FR ⊆ AH
⟨*proof*⟩

Finite range functions valued in an abelian group form a normal subgroup of almost homomorphisms.

lemma Group˙ZF˙3˙3˙L2: assumes A1:IsAgroup(G,f)
and A2:f –is commutative on″ G
shows
IsAsubgroup(FinRangeFunctions(G,G),AlHomOp1(G,f))
IsAnormalSubgroup(AlmostHoms(G,f),AlHomOp1(G,f),
FinRangeFunctions(G,G))
⟨*proof*⟩

The group of almost homomorphisms divided by the subgroup of finite range functions is an abelian group.

theorem (in group1) Group˙ZF˙3˙3˙T1:
shows
IsAgroup(AH//QuotientGroupRel(AH,Op1,FR),QuotientGroupOp(AH,Op1,FR))

and
QuotientGroupOp(AH,Op1,FR) {is commutative on}
(AH//QuotientGroupRel(AH,Op1,FR))
⟨*proof*⟩

It is useful to have a direct statement that the quotient group relation is an equivalence relation for the group of AH and subgroup FR.

lemma (in group1) Group_ZF_3_3_L3: shows
QuotientGroupRel(AH,Op1,FR) ⊆ AH × AH and
equiv(AH,QuotientGroupRel(AH,Op1,FR))
⟨*proof*⟩

The "almost equal" relation is symmetric.

lemma (in group1) Group_ZF_3_3_L3A: assumes A1: s≈r
shows r≈s
⟨*proof*⟩

Although we have bypassed this fact when proving that group of almost homomorphisms divided by the subgroup of finite range functions is a group, it is still useful to know directly that the first group operation on AH is congruent with respect to the quotient group relation.

lemma (in group1) Group_ZF_3_3_L4:
shows Congruent2(QuotientGroupRel(AH,Op1,FR),Op1)
⟨*proof*⟩

The class of an almost homomorphism s is the neutral element of the quotient group of almost homomorphisms iff s is a finite range function.

lemma (in group1) Group_ZF_3_3_L5: assumes s ∈ AH and
r = QuotientGroupRel(AH,Op1,FR) and
TheNeutralElement(AH//r,QuotientGroupOp(AH,Op1,FR)) = e
shows r{s} = e ⟷ s ∈ FR
⟨*proof*⟩

The group inverse of a class of an almost homomorphism f is the class of the inverse of f.

lemma (in group1) Group_ZF_3_3_L6:
assumes A1: s ∈ AH and
r = QuotientGroupRel(AH,Op1,FR) and
F = ProjFun2(AH,r,Op1)
shows r{∼s} = GroupInv(AH//r,F)(r{s})
⟨*proof*⟩

31.4 Compositions of almost homomorphisms

The goal of this section is to establish some facts about composition of almost homomorphisms. needed for the real numbers construction in Real_ZF_x series. In particular we show that the set of almost homomorphisms is closed under composition and that composition is congruent with respect to the equivalence relation defined by the group of finite range functions (a normal subgroup of almost homomorphisms).

The next formula restates the definition of the homomorphism difference to express the value an almost homomorphism on a product.

lemma (in group1) Group˙ZF˙3˙4˙L1:
  assumes s∈AH and m∈G n∈G
  shows $s(m \cdot n) = s(m) \cdot s(n) \cdot \delta(s, \langle m,n \rangle)$
  ⟨*proof*⟩

What is the value of a composition of almost homomorhisms?

lemma (in group1) Group˙ZF˙3˙4˙L2:
  assumes s∈AH r∈AH and m∈G
  shows $(s \circ r)(m) = s(r(m))$ $s(r(m)) \in G$
  ⟨*proof*⟩

What is the homomorphism difference of a composition?

lemma (in group1) Group˙ZF˙3˙4˙L3:
  assumes A1: s∈AH r∈AH and A2: m∈G n∈G
  shows $\delta(s \circ r, \langle m,n \rangle) =$
  $\delta(s, \langle r(m), r(n) \rangle) \cdot s(\delta(r, \langle m,n \rangle)) \cdot \delta(s, \langle r(m) \cdot r(n), \delta(r, \langle m,n \rangle) \rangle)$
⟨*proof*⟩

What is the homomorphism difference of a composition (another form)? Here we split the homomorphism difference of a composition into a product of three factors. This will help us in proving that the range of homomorphism difference for the composition is finite, as each factor has finite range.

lemma (in group1) Group˙ZF˙3˙4˙L4:
  assumes A1: s∈AH r∈AH and A2: $x \in G \times G$
  and A3:
  $A = \delta(s, \langle r(fst(x)), r(snd(x)) \rangle)$
  $B = s(\delta(r,x))$
  $C = \delta(s, \langle (r(fst(x)) \cdot r(snd(x))), \delta(r,x) \rangle)$
  shows $\delta(s \circ r, x) = A \cdot B \cdot C$
⟨*proof*⟩

The range of the homomorphism difference of a composition of two almost homomorphisms is finite. This is the essential condition to show that a composition of almost homomorphisms is an almost homomorphism.

lemma (in group1) Group˙ZF˙3˙4˙L5:
  assumes A1: s∈AH r∈AH
  shows –δ(Composition(G)⟨ s,r⟩,x). $x \in G \times G'' \in$ Fin(G)
⟨*proof*⟩

Composition of almost homomorphisms is an almost homomorphism.

theorem (in group1) Group˙ZF˙3˙4˙T1:

assumes A1: s∈AH r∈AH
shows Composition(G)⟨ s,r⟩ ∈ AH s∘r ∈ AH
⟨*proof*⟩

The set of almost homomorphisms is closed under composition. The second operation on almost homomorphisms is associative.

lemma (in group1) Group˙ZF˙3˙4˙L6: shows
AH –is closed under″ Composition(G)
AlHomOp2(G,P) –is associative on″ AH
⟨*proof*⟩

Type information related to the situation of two almost homomorphisms.

lemma (in group1) Group˙ZF˙3˙4˙L7:
assumes A1: s∈AH r∈AH and A2: n∈G
shows
s(n) ∈ G $(r(n))^{-1}$ ∈ G
s(n)·$(r(n))^{-1}$ ∈ G   s(r(n)) ∈ G
⟨*proof*⟩

Type information related to the situation of three almost homomorphisms.

lemma (in group1) Group˙ZF˙3˙4˙L8:
assumes A1: s∈AH r∈AH q∈AH and A2: n∈G
shows
q(n)∈G
s(r(n)) ∈ G
r(n)·$(q(n))^{-1}$ ∈ G
s(r(n)·$(q(n))^{-1}$) ∈ G
δ(s,⟨ q(n),r(n)·$(q(n))^{-1}$⟩) ∈ G
⟨*proof*⟩

A formula useful in showing that the composition of almost homomorphisms is congruent with respect to the quotient group relation.

lemma (in group1) Group˙ZF˙3˙4˙L9:
assumes A1: s1 ∈ AH r1 ∈ AH s2 ∈ AH r2 ∈ AH
and A2: n∈G
shows (s1∘r1)(n)·$((s2∘r2)(n))^{-1}$ =
s1(r2(n))· $(s2(r2(n)))^{-1}$·s1(r1(n)·$(r2(n))^{-1}$)·
δ(s1,⟨ r2(n),r1(n)·$(r2(n))^{-1}$⟩)
⟨*proof*⟩

The next lemma shows a formula that translates an expression in terms of the first group operation on almost homomorphisms and the group inverse in the group of almost homomorphisms to an expression using only the underlying group operations.

lemma (in group1) Group˙ZF˙3˙4˙L10: assumes A1: s ∈ AH r ∈ AH
and A2: n ∈ G
shows (s•(GroupInv(AH,Op1)(r)))(n) = s(n)·$(r(n))^{-1}$

⟨*proof*⟩

A neccessary condition for two a. h. to be almost equal.

lemma (in group1) Group˙ZF˙3˙4˙L11:
 assumes A1: s≈r
 shows –s(n)·(r(n))$^{-1}$. n∈G″ ∈ Fin(G)
⟨*proof*⟩

A sufficient condition for two a. h. to be almost equal.

lemma (in group1) Group˙ZF˙3˙4˙L12: assumes A1: s∈AH r∈AH
 and A2: –s(n)·(r(n))$^{-1}$. n∈G″ ∈ Fin(G)
 shows s≈r
⟨*proof*⟩

Another sufficient consdition for two a.h. to be almost equal. It is actually just an expansion of the definition of the quotient group relation.

lemma (in group1) Group˙ZF˙3˙4˙L12A: assumes s∈AH r∈AH
 and s•(GroupInv(AH,Op1)(r)) ∈ FR
 shows s≈r r≈s
⟨*proof*⟩

Another necessary condition for two a.h. to be almost equal. It is actually just an expansion of the definition of the quotient group relation.

lemma (in group1) Group˙ZF˙3˙4˙L12B: assumes s≈r
 shows s•(GroupInv(AH,Op1)(r)) ∈ FR
 ⟨*proof*⟩

The next lemma states the essential condition for the composition of a. h. to be congruent with respect to the quotient group relation for the subgroup of finite range functions.

lemma (in group1) Group˙ZF˙3˙4˙L13:
 assumes A1: s1≈s2 r1≈r2
 shows (s1∘r1) ≈ (s2∘r2)
⟨*proof*⟩

Composition of a. h. to is congruent with respect to the quotient group relation for the subgroup of finite range functions. Recall that if an operation say "∘" on X is congruent with respect to an equivalence relation R then we can define the operation on the quotient space X/R by $[s]_R \circ [r]_R := [s \circ r]_R$ and this definition will be correct i.e. it will not depend on the choice of representants for the classes $[x]$ and $[y]$. This is why we want it here.

lemma (in group1) Group˙ZF˙3˙4˙L13A: shows
 Congruent2(QuotientGroupRel(AH,Op1,FR),Op2)
⟨*proof*⟩

The homomorphism difference for the identity function is equal to the neutral element of the group (denoted e in the group1 context).

lemma (in group1) Group˙ZF˙3˙4˙L14: assumes A1: x ∈ G×G
shows δ(id(G),x) = **1**
⟨*proof*⟩

The identity function ($I(x) = x$) on G is an almost homomorphism.

lemma (in group1) Group˙ZF˙3˙4˙L15: shows id(G) ∈ AH
⟨*proof*⟩

Almost homomorphisms form a monoid with composition. The identity function on the group is the neutral element there.

lemma (in group1) Group˙ZF˙3˙4˙L16:
shows
IsAmonoid(AH,Op2)
monoid0(AH,Op2)
id(G) = TheNeutralElement(AH,Op2)
⟨*proof*⟩

We can project the monoid of almost homomorphisms with composition to the group of almost homomorphisms divided by the subgroup of finite range functions. The class of the identity function is the neutral element of the quotient (monoid).

theorem (in group1) Group˙ZF˙3˙4˙T2:
assumes A1: R = QuotientGroupRel(AH,Op1,FR)
shows
IsAmonoid(AH//R,ProjFun2(AH,R,Op2))
R–id(G)″ = TheNeutralElement(AH//R,ProjFun2(AH,R,Op2))
⟨*proof*⟩

31.5 Shifting almost homomorphisms

In this this section we consider what happens if we multiply an almost homomorphism by a group element. We show that the resulting function is also an a. h., and almost equal to the original one. This is used only for slopes (integer a.h.) in Int˙ZF˙2 where we need to correct a positive slopes by adding a constant, so that it is at least 2 on positive integers.

If s is an almost homomorphism and c is some constant from the group, then $s \cdot c$ is an almost homomorphism.

lemma (in group1) Group˙ZF˙3˙5˙L1:
assumes A1: s ∈ AH and A2: c∈G and
A3: r = –⟨x,s(x)·c⟩. x∈G″
shows
∀x∈G. r(x) = s(x)·c
r ∈ AH
s ≈ r
⟨*proof*⟩

end

32 Direct product

theory DirectProduct˙ZF imports func˙ZF

begin

This theory considers the direct product of binary operations. Contributed by Seo Sanghyeon.

32.1 Definition

In group theory the notion of direct product provides a natural way of creating a new group from two given groups.

Given $(G, \cdot)$ and $(H, \circ)$ a new operation $(G \times H, \times)$ is defined as $(g, h) \times (g', h') = (g \cdot g', h \circ h')$.

definition
  DirectProduct(P,Q,G,H) ≡
  –⟨x,⟨P⟨fst(fst(x)),fst(snd(x))⟩ , Q⟨snd(fst(x)),snd(snd(x))⟩⟩⟩.
  x ∈ (G×H)×(G×H)″

We define a context called direct0 which holds an assumption that P, Q are binary operations on G, H, resp. and denotes R as the direct product of (G, P) and (H, Q).

locale direct0 =
  fixes P Q G H
  assumes Pfun: P : G×G→G
  assumes Qfun: Q : H×H→H
  fixes R
  defines Rdef [simp]: R ≡ DirectProduct(P,Q,G,H)

The direct product of binary operations is a binary operation.

lemma (in direct0) DirectProduct˙ZF˙1˙L1:
  shows R : (G×H)×(G×H)→G×H
⟨*proof*⟩

And it has the intended value.

lemma (in direct0) DirectProduct˙ZF˙1˙L2:
  shows ∀x∈(G×H). ∀y∈(G×H).
  R⟨x,y⟩ = ⟨P⟨fst(x),fst(y)⟩,Q⟨snd(x),snd(y)⟩⟩
  ⟨*proof*⟩

And the value belongs to the set the operation is defined on.

lemma (in direct0) DirectProduct˙ZF˙1˙L3:
  shows ∀x∈(G×H). ∀y∈(G×H). R⟨x,y⟩ ∈ G×H
  ⟨*proof*⟩

32.2 Associative and commutative operations

If P and Q are both associative or commutative operations, the direct product of P and Q has the same property.

Direct product of commutative operations is commutative.

lemma (in direct0) DirectProduct˙ZF˙2˙L1:
  assumes P –is commutative on˝ G and Q –is commutative on˝ H
  shows R –is commutative on˝ G×H
⟨*proof*⟩

Direct product of associative operations is associative.

lemma (in direct0) DirectProduct˙ZF˙2˙L2:
  assumes P –is associative on˝ G and Q –is associative on˝ H
  shows R –is associative on˝ G×H
⟨*proof*⟩

end

33 Ordered groups - introduction

theory OrderedGroup˙ZF imports Group˙ZF˙1 AbelianGroup˙ZF Order˙ZF Finite˙ZF˙1

begin

This theory file defines and shows the basic properties of (partially or linearly) ordered groups. We define the set of nonnegative elements and the absolute value function. We show that in linearly ordered groups finite sets are bounded and provide a sufficient condition for bounded sets to be finite. This allows to show in Int˙ZF˙IML.thy that subsets of integers are bounded iff they are finite.

33.1 Ordered groups

This section defines ordered groups and various related notions.

An ordered group is a group equipped with a partial order that is "translation invariant", that is if $a \leq b$ then $a \cdot g \leq b \cdot g$ and $g \cdot a \leq g \cdot b$.

definition
  IsAnOrdGroup(G,P,r) ≡
  (IsAgroup(G,P) ∧ r⊆G×G ∧ IsPartOrder(G,r) ∧ (∀g∈G. ∀a b.
  ⟨ a,b⟩ ∈ r ⟶ ⟨ P⟨ a,g⟩,P⟨ b,g⟩ ⟩ ∈ r ∧ ⟨ P⟨ g,a⟩,P⟨ g,b⟩ ⟩ ∈ r))

We define the set of nonnegative elements in the obvious way as $G^+ = \{x \in G : 1 \leq x\}$.

definition

Nonnegative(G,P,r) ≡ –x∈G. ⟨ TheNeutralElement(G,P),x⟩ ∈ r″

The PositiveSet(G,P,r) is a set similar to Nonnegative(G,P,r), but without the unit.

definition
PositiveSet(G,P,r) ≡
–x∈G. ⟨ TheNeutralElement(G,P),x⟩ ∈ r ∧ TheNeutralElement(G,P)≠ x″

We also define the absolute value as a ZF-function that is the identity on G^+ and the group inverse on the rest of the group.

definition
AbsoluteValue(G,P,r) ≡ id(Nonnegative(G,P,r)) ∪
restrict(GroupInv(G,P),G - Nonnegative(G,P,r))

The odd functions are defined as those having property $f(a^{-1}) = (f(a))^{-1}$. This looks a bit strange in the multiplicative notation, I have to admit. For linearly oredered groups a function f defined on the set of positive elements iniquely defines an odd function of the whole group. This function is called an odd extension of f

definition
OddExtension(G,P,r,f) ≡
(f ∪ –⟨a, GroupInv(G,P)(f(GroupInv(G,P)(a)))⟩.
a ∈ GroupInv(G,P)(PositiveSet(G,P,r))″ ∪
–⟨TheNeutralElement(G,P),TheNeutralElement(G,P)⟩″)

We will use a similar notation for ordered groups as for the generic groups. G$^+$ denotes the set of nonnegative elements (that satisfy $1 \leq a$) and G$_+$ is the set of (strictly) positive elements. -A is the set inverses of elements from A. I hope that using additive notation for this notion is not too shocking here. The symbol f$^\circ$ denotes the odd extension of f. For a function defined on G_+ this is the unique odd function on G that is equal to f on G_+.

locale group3 =

fixes G and P and r

assumes ordGroupAssum: IsAnOrdGroup(G,P,r)

fixes unit (**1**)
defines unit˙def [simp]: **1** ≡ TheNeutralElement(G,P)

fixes groper (infixl · 70)
defines groper˙def [simp]: a · b ≡ P⟨ a,b⟩

fixes inv ($\cdot^{-1}$ [90] 91)
defines inv˙def [simp]: x^{-1} ≡ GroupInv(G,P)(x)

fixes lesseq (infix ≤ 68)

defines lesseq˙def [simp]: a ≤ b ≡ ⟨ a,b⟩ ∈ r

fixes sless (infix ¡ 68)
defines sless˙def [simp]: a ¡ b ≡ a≤b ∧ a≠b

fixes nonnegative (G^+)
defines nonnegative˙def [simp]: G^+ ≡ Nonnegative(G,P,r)

fixes positive (G_+)
defines positive˙def [simp]: G_+ ≡ PositiveSet(G,P,r)

fixes setinv (- ˙ 72)
defines setninv˙def [simp]: -A ≡ GroupInv(G,P)(A)

fixes abs (— ˙ —)
defines abs˙def [simp]: —a— ≡ AbsoluteValue(G,P,r)(a)

fixes oddext (˙ °)
defines oddext˙def [simp]: f° ≡ OddExtension(G,P,r,f)

In group3 context we can use the theorems proven in the group0 context.

lemma (in group3) OrderedGroup˙ZF˙1˙L1: shows group0(G,P)
⟨*proof*⟩

Ordered group (carrier) is not empty. This is a property of monoids, but it is good to have it handy in the group3 context.

lemma (in group3) OrderedGroup˙ZF˙1˙L1A: shows G≠0
⟨*proof*⟩

The next lemma is just to see the definition of the nonnegative set in our notation.

lemma (in group3) OrderedGroup˙ZF˙1˙L2:
shows g∈G^+ ⟷ **1**≤g
⟨*proof*⟩

The next lemma is just to see the definition of the positive set in our notation.

lemma (in group3) OrderedGroup˙ZF˙1˙L2A:
shows g∈G_+ ⟷ (**1**≤g ∧ g≠**1**)
⟨*proof*⟩

For total order if g is not in G^+, then it has to be less or equal the unit.

lemma (in group3) OrderedGroup˙ZF˙1˙L2B:
assumes A1: r –is total on″ G and A2: a∈G-G^+
shows a≤**1**
⟨*proof*⟩

The group order is reflexive.

lemma (in group3) OrderedGroup˙ZF˙1˙L3: assumes g∈G

shows g≤g
⟨*proof*⟩

1 is nonnegative.

lemma (in group3) OrderedGroup˙ZF˙1˙L3A: shows **1**∈G^{+}
⟨*proof*⟩

In this context $a \leq b$ implies that both a and b belong to G.

lemma (in group3) OrderedGroup˙ZF˙1˙L4:
assumes a≤b shows a∈G b∈G
⟨*proof*⟩

Similarly in this context $a \leq b$ implies that both a and b belong to G.

lemma (in group3) less˙are˙members:
assumes a¡b shows a∈G b∈G
⟨*proof*⟩

It is good to have transitivity handy.

lemma (in group3) Group˙order˙transitive:
assumes A1: a≤b b≤c shows a≤c
⟨*proof*⟩

The order in an ordered group is antisymmetric.

lemma (in group3) group˙order˙antisym:
assumes A1: a≤b b≤a shows a=b
⟨*proof*⟩

Transitivity for the strict order: if $a < b$ and $b \leq c$, then $a < c$.

lemma (in group3) OrderedGroup˙ZF˙1˙L4A:
assumes A1: a¡b and A2: b≤c
shows a¡c
⟨*proof*⟩

Another version of transitivity for the strict order: if $a \leq b$ and $b < c$, then $a < c$.

lemma (in group3) group˙strict˙ord˙transit:
assumes A1: a≤b and A2: b¡c
shows a¡c
⟨*proof*⟩

Strict order is preserved by translations.

lemma (in group3) group˙strict˙ord˙transl˙inv:
assumes a¡b and c∈G
shows
a·c ¡ b·c
c·a ¡ c·b
⟨*proof*⟩

If the group order is total, then the group is ordered linearly.

lemma (in group3) group_ord_total_is_lin:
  assumes r {is total on} G
  shows IsLinOrder(G,r)
  ⟨*proof*⟩

For linearly ordered groups elements in the nonnegative set are greater than those in the complement.

lemma (in group3) OrderedGroup_ZF_1_L4B:
  assumes r {is total on} G
  and a∈G^+ and b ∈ G-G^+
  shows b≤a
⟨*proof*⟩

If $a \leq 1$ and $a \neq 1$, then $a \in G \setminus G^+$.

lemma (in group3) OrderedGroup_ZF_1_L4C:
  assumes A1: a≤**1** and A2: a≠**1**
  shows a ∈ G-G^+
⟨*proof*⟩

An element smaller than an element in $G \setminus G^+$ is in $G \setminus G^+$.

lemma (in group3) OrderedGroup_ZF_1_L4D:
  assumes A1: a∈G-G^+ and A2: b≤a
  shows b∈G-G^+
⟨*proof*⟩

The nonnegative set is contained in the group.

lemma (in group3) OrderedGroup_ZF_1_L4E: shows G^+ ⊆ G
  ⟨*proof*⟩

Taking the inverse on both sides reverses the inequality.

lemma (in group3) OrderedGroup_ZF_1_L5:
  assumes A1: a≤b shows b^{-1}≤a^{-1}
⟨*proof*⟩

If an element is smaller that the unit, then its inverse is greater.

lemma (in group3) OrderedGroup_ZF_1_L5A:
  assumes A1: a≤**1** shows **1**≤a^{-1}
⟨*proof*⟩

If an the inverse of an element is greater that the unit, then the element is smaller.

lemma (in group3) OrderedGroup_ZF_1_L5AA:
  assumes A1: a∈G and A2: **1**≤a^{-1}
  shows a≤**1**
⟨*proof*⟩

If an element is nonnegative, then the inverse is not greater that the unit. Also shows that nonnegative elements cannot be negative

lemma (in group3) OrderedGroup_ZF_1_L5AB:
  assumes A1: $\mathbf{1} \leq a$ shows $a^{-1} \leq \mathbf{1}$ and $\neg(a \leq \mathbf{1} \wedge a \neq \mathbf{1})$
⟨*proof*⟩

If two elements are greater or equal than the unit, then the inverse of one is not greater than the other.

lemma (in group3) OrderedGroup_ZF_1_L5AC:
  assumes A1: $\mathbf{1} \leq a$ $\mathbf{1} \leq b$
  shows $a^{-1} \leq b$
⟨*proof*⟩

33.2 Inequalities

This section developes some simple tools to deal with inequalities.

Taking negative on both sides reverses the inequality, case with an inverse on one side.

lemma (in group3) OrderedGroup_ZF_1_L5AD:
  assumes A1: $b \in G$ and A2: $a \leq b^{-1}$
  shows $b \leq a^{-1}$
⟨*proof*⟩

We can cancel the same element on both sides of an inequality.

lemma (in group3) OrderedGroup_ZF_1_L5AE:
  assumes A1: $a \in G$ $b \in G$ $c \in G$ and A2: $a \cdot b \leq a \cdot c$
  shows $b \leq c$
⟨*proof*⟩

We can cancel the same element on both sides of an inequality, a version with an inverse on both sides.

lemma (in group3) OrderedGroup_ZF_1_L5AF:
  assumes A1: $a \in G$ $b \in G$ $c \in G$ and A2: $a \cdot b^{-1} \leq a \cdot c^{-1}$
  shows $c \leq b$
⟨*proof*⟩

Taking negative on both sides reverses the inequality, another case with an inverse on one side.

lemma (in group3) OrderedGroup_ZF_1_L5AG:
  assumes A1: $a \in G$ and A2: $a^{-1} \leq b$
  shows $b^{-1} \leq a$
⟨*proof*⟩

We can multiply the sides of two inequalities.

lemma (in group3) OrderedGroup_ZF_1_L5B:

assumes A1: a≤b and A2: c≤d
shows a·c ≤ b·d
⟨*proof*⟩

We can replace first of the factors on one side of an inequality with a greater one.

lemma (in group3) OrderedGroup˙ZF˙1˙L5C:
assumes A1: c∈G and A2: a≤b·c and A3: b≤b_1
shows a≤b_1·c
⟨*proof*⟩

We can replace second of the factors on one side of an inequality with a greater one.

lemma (in group3) OrderedGroup˙ZF˙1˙L5D:
assumes A1: b∈G and A2: a ≤ b·c and A3: c≤b_1
shows a ≤ b·b_1
⟨*proof*⟩

We can replace factors on one side of an inequality with greater ones.

lemma (in group3) OrderedGroup˙ZF˙1˙L5E:
assumes A1: a ≤ b·c and A2: b≤b_1 c≤c_1
shows a ≤ b_1·c_1
⟨*proof*⟩

We don't decrease an element of the group by multiplying by one that is nonnegative.

lemma (in group3) OrderedGroup˙ZF˙1˙L5F:
assumes A1: **1**≤a and A2: b∈G
shows b≤a·b b≤b·a
⟨*proof*⟩

We can multiply the right hand side of an inequality by a nonnegative element.

lemma (in group3) OrderedGroup˙ZF˙1˙L5G: assumes A1: a≤b
and A2: **1**≤c shows a≤b·c a≤c·b
⟨*proof*⟩

We can put two elements on the other side of inequality, changing their sign.

lemma (in group3) OrderedGroup˙ZF˙1˙L5H:
assumes A1: a∈G b∈G and A2: a·b^{-1} ≤ c
shows
a ≤ c·b
c^{-1}·a ≤ b
⟨*proof*⟩

We can multiply the sides of one inequality by inverse of another.

lemma (in group3) OrderedGroup˙ZF˙1˙L5I:

assumes a≤b and c≤d
shows $a \cdot d^{-1} \leq b \cdot c^{-1}$
⟨*proof*⟩

We can put an element on the other side of an inequality changing its sign, version with the inverse.

lemma (in group3) OrderedGroup_ZF_1_L5J:
assumes A1: a∈G b∈G and A2: $c \leq a \cdot b^{-1}$
shows $c \cdot b \leq a$
⟨*proof*⟩

We can put an element on the other side of an inequality changing its sign, version with the inverse.

lemma (in group3) OrderedGroup_ZF_1_L5JA:
assumes A1: a∈G b∈G and A2: $c \leq a^{-1} \cdot b$
shows $a \cdot c \leq b$
⟨*proof*⟩

A special case of OrderedGroup_ZF_1_L5J where $c = 1$.

corollary (in group3) OrderedGroup_ZF_1_L5K:
assumes A1: a∈G b∈G and A2: $\mathbf{1} \leq a \cdot b^{-1}$
shows b ≤ a
⟨*proof*⟩

A special case of OrderedGroup_ZF_1_L5JA where $c = 1$.

corollary (in group3) OrderedGroup_ZF_1_L5KA:
assumes A1: a∈G b∈G and A2: $\mathbf{1} \leq a^{-1} \cdot b$
shows a ≤ b
⟨*proof*⟩

If the order is total, the elements that do not belong to the positive set are negative. We also show here that the group inverse of an element that does not belong to the nonnegative set does belong to the nonnegative set.

lemma (in group3) OrderedGroup_ZF_1_L6:
assumes A1: r {is total on} G and A2: $a \in G - G^+$
shows $a \leq \mathbf{1}$ $a^{-1} \in G^+$ $\text{restrict}(\text{GroupInv}(G,P),G-G^+)(a) \in G^+$
⟨*proof*⟩

If a property is invariant with respect to taking the inverse and it is true on the nonnegative set, than it is true on the whole group.

lemma (in group3) OrderedGroup_ZF_1_L7:
assumes A1: r {is total on} G
and A2: $\forall a \in G^+. \forall b \in G^+.\ Q(a,b)$
and A3: $\forall a \in G. \forall b \in G.\ Q(a,b) \longrightarrow Q(a^{-1},b)$
and A4: $\forall a \in G. \forall b \in G.\ Q(a,b) \longrightarrow Q(a,b^{-1})$
and A5: a∈G b∈G
shows Q(a,b)

⟨*proof*⟩

A lemma about splitting the ordered group "plane" into 6 subsets. Useful for proofs by cases.

lemma (in group3) OrdGroup˙6cases: assumes A1: r –is total on" G
and A2: a∈G b∈G
shows
$\mathbf{1}$≤a ∧ $\mathbf{1}$≤b ∨ a≤$\mathbf{1}$ ∧ b≤$\mathbf{1}$ ∨
a≤$\mathbf{1}$ ∧ $\mathbf{1}$≤b ∧ $\mathbf{1}$ ≤ a·b ∨ a≤$\mathbf{1}$ ∧ $\mathbf{1}$≤b ∧ a·b ≤ $\mathbf{1}$ ∨
$\mathbf{1}$≤a ∧ b≤$\mathbf{1}$ ∧ $\mathbf{1}$ ≤ a·b ∨ $\mathbf{1}$≤a ∧ b≤$\mathbf{1}$ ∧ a·b ≤ $\mathbf{1}$
⟨*proof*⟩

The next lemma shows what happens when one element of a totally ordered group is not greater or equal than another.

lemma (in group3) OrderedGroup˙ZF˙1˙L8:
assumes A1: r –is total on" G
and A2: a∈G b∈G
and A3: ¬(a≤b)
shows b ≤ a a^{-1} ≤ b^{-1} a≠b b¡a

⟨*proof*⟩

If one element is greater or equal and not equal to another, then it is not smaller or equal.

lemma (in group3) OrderedGroup˙ZF˙1˙L8AA:
assumes A1: a≤b and A2: a≠b
shows ¬(b≤a)
⟨*proof*⟩

A special case of OrderedGroup˙ZF˙1˙L8 when one of the elements is the unit.

corollary (in group3) OrderedGroup˙ZF˙1˙L8A:
assumes A1: r –is total on" G
and A2: a∈G and A3: ¬($\mathbf{1}$≤a)
shows $\mathbf{1}$ ≤ a^{-1} $\mathbf{1}$≠a a≤$\mathbf{1}$
⟨*proof*⟩

A negative element can not be nonnegative.

lemma (in group3) OrderedGroup˙ZF˙1˙L8B:
assumes A1: a≤$\mathbf{1}$ and A2: a≠$\mathbf{1}$ shows ¬($\mathbf{1}$≤a)
⟨*proof*⟩

An element is greater or equal than another iff the difference is nonpositive.

lemma (in group3) OrderedGroup˙ZF˙1˙L9:
assumes A1: a∈G b∈G
shows a≤b ⟷ a·b^{-1} ≤ $\mathbf{1}$
⟨*proof*⟩

We can move an element to the other side of an inequality.

lemma (in group3) OrderedGroup_ZF_1_L9A:
  assumes A1: a∈G b∈G c∈G
  shows a·b ≤ c ⟷ a ≤ c·b^{-1}
⟨*proof*⟩

A one side version of the previous lemma with weaker assuptions.

lemma (in group3) OrderedGroup_ZF_1_L9B:
  assumes A1: a∈G b∈G and A2: a·b^{-1} ≤ c
  shows a ≤ c·b
⟨*proof*⟩

We can put en element on the other side of inequality, changing its sign.

lemma (in group3) OrderedGroup_ZF_1_L9C:
  assumes A1: a∈G b∈G and A2: c≤a·b
  shows
  c·b^{-1} ≤ a
  a^{-1}·c ≤ b
⟨*proof*⟩

If an element is greater or equal than another then the difference is nonnegative.

lemma (in group3) OrderedGroup_ZF_1_L9D: assumes A1: a≤b
  shows **1** ≤ b·a^{-1}
⟨*proof*⟩

If an element is greater than another then the difference is positive.

lemma (in group3) OrderedGroup_ZF_1_L9E:
  assumes A1: a≤b a≠b
  shows **1** ≤ b·a^{-1} **1** ≠ b·a^{-1} b·a^{-1} ∈ G$_+$
⟨*proof*⟩

If the difference is nonnegative, then $a \leq b$.

lemma (in group3) OrderedGroup_ZF_1_L9F:
  assumes A1: a∈G b∈G and A2: **1** ≤ b·a^{-1}
  shows a≤b
⟨*proof*⟩

If we increase the middle term in a product, the whole product increases.

lemma (in group3) OrderedGroup_ZF_1_L10:
  assumes a∈G b∈G and c≤d
  shows a·c·b ≤ a·d·b
  ⟨*proof*⟩

A product of (strictly) positive elements is not the unit.

lemma (in group3) OrderedGroup_ZF_1_L11:
  assumes A1: **1**≤a **1**≤b
  and A2: **1** ≠ a **1** ≠ b

shows $\mathbf{1} \neq$ a·b
$\langle proof \rangle$

A product of nonnegative elements is nonnegative.

lemma (in group3) OrderedGroup˙ZF˙1˙L12:
assumes A1: $\mathbf{1} \leq$ a $\mathbf{1} \leq$ b
shows $\mathbf{1} \leq$ a·b
$\langle proof \rangle$

If a is not greater than b, then 1 is not greater than $b \cdot a^{-1}$.

lemma (in group3) OrderedGroup˙ZF˙1˙L12A:
assumes A1: a≤b shows $\mathbf{1} \leq$ b·a^{-1}
$\langle proof \rangle$

We can move an element to the other side of a strict inequality.

lemma (in group3) OrderedGroup˙ZF˙1˙L12B:
assumes A1: a∈G b∈G and A2: a·b^{-1} ¡ c
shows a ¡ c·b
$\langle proof \rangle$

We can multiply the sides of two inequalities, first of them strict and we get a strict inequality.

lemma (in group3) OrderedGroup˙ZF˙1˙L12C:
assumes A1: a¡b and A2: c≤d
shows a·c ¡ b·d
$\langle proof \rangle$

We can multiply the sides of two inequalities, second of them strict and we get a strict inequality.

lemma (in group3) OrderedGroup˙ZF˙1˙L12D:
assumes A1: a≤b and A2: c¡d
shows a·c ¡ b·d
$\langle proof \rangle$

33.3 The set of positive elements

In this section we study G_+ - the set of elements that are (strictly) greater than the unit. The most important result is that every linearly ordered group can decomposed into $\{1\}$, G_+ and the set of those elements $a \in G$ such that a^{-1} $\in G_+$. Another property of linearly ordered groups that we prove here is that if $G_+ \neq \emptyset$, then it is infinite. This allows to show that nontrivial linearly ordered groups are infinite.

The positive set is closed under the group operation.

lemma (in group3) OrderedGroup˙ZF˙1˙L13: shows G_+ –is closed under˝ P
$\langle proof \rangle$

For totally ordered groups every nonunit element is positive or its inverse is positive.

lemma (in group3) OrderedGroup_ZF_1_L14:
assumes A1: r {is total on} G and A2: a∈G
shows a=**1** ∨ a∈G_+ ∨ a^{-1}∈G_+
⟨*proof*⟩

If an element belongs to the positive set, then it is not the unit and its inverse does not belong to the positive set.

lemma (in group3) OrderedGroup_ZF_1_L15:
assumes A1: a∈G_+ shows a≠**1** a^{-1}∉G_+
⟨*proof*⟩

If a^{-1} is positive, then a can not be positive or the unit.

lemma (in group3) OrderedGroup_ZF_1_L16:
assumes A1: a∈G and A2: a^{-1}∈G_+ shows a≠**1** a∉G_+
⟨*proof*⟩

For linearly ordered groups each element is either the unit, positive or its inverse is positive.

lemma (in group3) OrdGroup_decomp:
assumes A1: r {is total on} G and A2: a∈G
shows Exactly_1_of_3_holds (a=**1**,a∈G_+,a^{-1}∈G_+)
⟨*proof*⟩

A if a is a nonunit element that is not positive, then a^{-1} is is positive. This is useful for some proofs by cases.

lemma (in group3) OrdGroup_cases:
assumes A1: r {is total on} G and A2: a∈G
and A3: a≠**1** a∉G_+
shows a^{-1} ∈ G_+
⟨*proof*⟩

Elements from $G \setminus G_+$ are not greater that the unit.

lemma (in group3) OrderedGroup_ZF_1_L17:
assumes A1: r {is total on} G and A2: a ∈ G-G_+
shows a≤**1**
⟨*proof*⟩

The next lemma allows to split proofs that something holds for all $a \in G$ into cases $a = 1$, $a \in G_+$, $-a \in G_+$.

lemma (in group3) OrderedGroup_ZF_1_L18:
assumes A1: r {is total on} G and A2: b∈G
and A3: Q(**1**) and A4: ∀a∈G_+. Q(a) and A5: ∀a∈G_+. Q(a^{-1})
shows Q(b)
⟨*proof*⟩

All elements greater or equal than an element of G_+ belong to G_+.

lemma (in group3) OrderedGroup_ZF_1_L19:
 assumes A1: $a \in G_+$ and A2: $a \leq b$
 shows $b \in G_+$
⟨*proof*⟩

The inverse of an element of G_+ cannot be in G_+.

lemma (in group3) OrderedGroup_ZF_1_L20:
 assumes A1: r {is total on} G and A2: $a \in G_+$
 shows $a^{-1} \notin G_+$
⟨*proof*⟩

The set of positive elements of a nontrivial linearly ordered group is not empty.

lemma (in group3) OrderedGroup_ZF_1_L21:
 assumes A1: r {is total on} G and A2: $G \neq \{\mathbf{1}\}$
 shows $G_+ \neq 0$
⟨*proof*⟩

If $b \in G_+$, then $a < a \cdot b$. Multiplying a by a positive elemnt increases a.

lemma (in group3) OrderedGroup_ZF_1_L22:
 assumes A1: $a\in G$ $b\in G_+$
 shows $a\leq a\cdot b$ $a \neq a\cdot b$ $a\cdot b \in G$
⟨*proof*⟩

If G is a nontrivial linearly ordered hroup, then for every element of G we can find one in G_+ that is greater or equal.

lemma (in group3) OrderedGroup_ZF_1_L23:
 assumes A1: r {is total on} G and A2: $G \neq \{\mathbf{1}\}$
 and A3: $a\in G$
 shows $\exists b\in G_+.\ a\leq b$
⟨*proof*⟩

The G^+ is G_+ plus the unit.

lemma (in group3) OrderedGroup_ZF_1_L24: shows $G^+ = G_+\cup\{\mathbf{1}\}$
 ⟨*proof*⟩

What is $-G_+$, really?

lemma (in group3) OrderedGroup_ZF_1_L25: shows
 $(\text{-}G_+) = \{a^{-1}.\ a\in G_+\}$
 $(\text{-}G_+) \subseteq G$
⟨*proof*⟩

If the inverse of a is in G_+, then a is in the inverse of G_+.

lemma (in group3) OrderedGroup_ZF_1_L26:
 assumes A1: $a\in G$ and A2: $a^{-1} \in G_+$
 shows $a \in (\text{-}G_+)$

⟨*proof*⟩

If a is in the inverse of G_+, then its inverse is in G_+.

lemma (in group3) OrderedGroup˙ZF˙1˙L27:
  assumes a ∈ (-G_+)
  shows a^{-1} ∈ G_+
  ⟨*proof*⟩

A linearly ordered group can be decomposed into G_+, $\{1\}$ and $-G_+$

lemma (in group3) OrdGroup˙decomp2:
  assumes A1: r –is total on˝ G
  shows
  G = G_+ ∪ (-G_+)∪ –**1**˝
  G_+∩(-G_+) = 0
  1 ∉ G_+∪(-G_+)
⟨*proof*⟩

If $a \cdot b^{-1}$ is nonnegative, then $b \leq a$. This maybe used to recover the order from the set of nonnegative elements and serve as a way to define order by prescibing that set (see the "Alternative definitions" section).

lemma (in group3) OrderedGroup˙ZF˙1˙L28:
  assumes A1: a∈G b∈G and A2: a·b^{-1} ∈ G^+
  shows b≤a
⟨*proof*⟩

A special case of OrderedGroup˙ZF˙1˙L28 when $a \cdot b^{-1}$ is positive.

corollary (in group3) OrderedGroup˙ZF˙1˙L29:
  assumes A1: a∈G b∈G and A2: a·b^{-1} ∈ G_+
  shows b≤a b≠a
⟨*proof*⟩

A bit stronger that OrderedGroup˙ZF˙1˙L29, adds case when two elements are equal.

lemma (in group3) OrderedGroup˙ZF˙1˙L30:
  assumes a∈G b∈G and a=b ∨ b·a^{-1} ∈ G_+
  shows a≤b
  ⟨*proof*⟩

A different take on decomposition: we can have $a = b$ or $a < b$ or $b < a$.

lemma (in group3) OrderedGroup˙ZF˙1˙L31:
  assumes A1: r –is total on˝ G and A2: a∈G b∈G
  shows a=b ∨ (a≤b ∧ a≠b) ∨ (b≤a ∧ b≠a)
⟨*proof*⟩

33.4 Intervals and bounded sets

Intervals here are the closed intervals of the form $\{x \in G.a \leq x \leq b\}$.

A bounded set can be translated to put it in G^+ and then it is still bounded above.

lemma (in group3) OrderedGroup˙ZF˙2˙L1:
  assumes A1: ∀g∈A. L≤g ∧ g≤M
  and A2: S = RightTranslation(G,P,L^{-1})
  and A3: a ∈ S(A)
  shows a ≤ M·L^{-1} **1**≤a
⟨*proof*⟩

Every bounded set is an image of a subset of an interval that starts at 1.

lemma (in group3) OrderedGroup˙ZF˙2˙L2:
  assumes A1: IsBounded(A,r)
  shows ∃B.∃g∈G^+.∃T∈G→G. A = T(B) ∧ B ⊆ Interval(r,**1**,g)
⟨*proof*⟩

If every interval starting at 1 is finite, then every bounded set is finite. I find it interesting that this does not require the group to be linearly ordered (the order to be total).

theorem (in group3) OrderedGroup˙ZF˙2˙T1:
  assumes A1: ∀g∈G^+. Interval(r,**1**,g) ∈ Fin(G)
  and A2: IsBounded(A,r)
  shows A ∈ Fin(G)
⟨*proof*⟩

In linearly ordered groups finite sets are bounded.

theorem (in group3) ord˙group˙fin˙bounded:
  assumes r –is total on″ G and B∈Fin(G)
  shows IsBounded(B,r)
  ⟨*proof*⟩

For nontrivial linearly ordered groups if for every element G we can find one in A that is greater or equal (not necessarily strictly greater), then A can neither be finite nor bounded above.

lemma (in group3) OrderedGroup˙ZF˙2˙L2A:
  assumes A1: r –is total on″ G and A2: G ≠ –**1**″
  and A3: ∀a∈G. ∃b∈A. a≤b
  shows
  ∀a∈G. ∃b∈A. a≠b ∧ a≤b
  ¬IsBoundedAbove(A,r)
  A ∉ Fin(G)
⟨*proof*⟩

Nontrivial linearly ordered groups are infinite. Recall that Fin(A) is the collection of finite subsets of A. In this lemma we show that $G \notin$ Fin(G), that is that G is not a finite subset of itself. This is a way of saying that G is infinite. We also show that for nontrivial linearly ordered groups G_+ is infinite.

theorem (in group3) Linord˙group˙infinite:
  assumes A1: r –is total on″ G and A2: G $\neq$ –**1**″
  shows
  $G_+ \notin$ Fin(G)
  G $\notin$ Fin(G)
⟨*proof*⟩

A property of nonempty subsets of linearly ordered groups that don't have a maximum: for any element in such subset we can find one that is strictly greater.

lemma (in group3) OrderedGroup˙ZF˙2˙L2B:
  assumes A1: r –is total on″ G and A2: A⊆G and
  A3: ¬HasAmaximum(r,A) and A4: x∈A
  shows ∃y∈A. x¡y
⟨*proof*⟩

In linearly ordered groups $G \setminus G_+$ is bounded above.

lemma (in group3) OrderedGroup˙ZF˙2˙L3:
  assumes A1: r –is total on″ G shows IsBoundedAbove(G-G_+,r)
⟨*proof*⟩

In linearly ordered groups if $A \cap G_+$ is finite, then A is bounded above.

lemma (in group3) OrderedGroup˙ZF˙2˙L4:
  assumes A1: r –is total on″ G and A2: A⊆G
  and A3: A ∩ G_+ ∈ Fin(G)
  shows IsBoundedAbove(A,r)
⟨*proof*⟩

If a set $-A \subseteq G$ is bounded above, then A is bounded below.

lemma (in group3) OrderedGroup˙ZF˙2˙L5:
  assumes A1: A⊆G and A2: IsBoundedAbove(-A,r)
  shows IsBoundedBelow(A,r)
⟨*proof*⟩

If $a \leq b$, then the image of the interval $a..b$ by any function is nonempty.

lemma (in group3) OrderedGroup˙ZF˙2˙L6:
  assumes a≤b and f:G→G
  shows f(Interval(r,a,b)) $\neq$ 0
  ⟨*proof*⟩

end

34 More on ordered groups

theory OrderedGroup˙ZF˙1 imports OrderedGroup˙ZF

begin

In this theory we continue the OrderedGroup˙ZF theory development.

34.1 Absolute value and the triangle inequality

The goal of this section is to prove the triangle inequality for ordered groups.

Absolute value maps G into G.

lemma (in group3) OrderedGroup˙ZF˙3˙L1:
 shows AbsoluteValue(G,P,r) : G→G
⟨*proof*⟩

If $a \in G^+$, then $|a| = a$.

lemma (in group3) OrderedGroup˙ZF˙3˙L2:
 assumes A1: a∈G^+ shows —a— = a
⟨*proof*⟩

The absolute value of the unit is the unit. In the additive totation that would be $|0| = 0$.

lemma (in group3) OrderedGroup˙ZF˙3˙L2A:
 shows —**1**— = **1** ⟨*proof*⟩

If a is positive, then $|a| = a$.

lemma (in group3) OrderedGroup˙ZF˙3˙L2B:
 assumes a∈G_+ shows —a— = a
 ⟨*proof*⟩

If $a \in G \setminus G^+$, then $|a| = a^{-1}$.

lemma (in group3) OrderedGroup˙ZF˙3˙L3:
 assumes A1: a ∈ G-G^+ shows —a— = a^{-1}
⟨*proof*⟩

For elements that not greater than the unit, the absolute value is the inverse.

lemma (in group3) OrderedGroup˙ZF˙3˙L3A:
 assumes A1: a≤**1**
 shows —a— = a^{-1}
⟨*proof*⟩

In linearly ordered groups the absolute value of any element is in G^+.

lemma (in group3) OrderedGroup˙ZF˙3˙L3B:
 assumes A1: r –is total on″ G and A2: a∈G
 shows —a— ∈ G^+
⟨*proof*⟩

For linearly ordered groups (where the order is total), the absolute value maps the group into the positive set.

lemma (in group3) OrderedGroup˙ZF˙3˙L3C:

assumes A1: r –is total on″ G
shows AbsoluteValue(G,P,r) : G→G$^+$
⟨*proof*⟩

If the absolute value is the unit, then the elemnent is the unit.

lemma (in group3) OrderedGroup˙ZF˙3˙L3D:
assumes A1: a∈G and A2: —a— = **1**
shows a = **1**
⟨*proof*⟩

In linearly ordered groups the unit is not greater than the absolute value of any element.

lemma (in group3) OrderedGroup˙ZF˙3˙L3E:
assumes r –is total on″ G and a∈G
shows **1** ≤ —a—
⟨*proof*⟩

If b is greater than both a and a^{-1}, then b is greater than $|a|$.

lemma (in group3) OrderedGroup˙ZF˙3˙L4:
assumes A1: a≤b and A2: a^{-1}≤ b
shows —a—≤ b
⟨*proof*⟩

In linearly ordered groups $a \leq |a|$.

lemma (in group3) OrderedGroup˙ZF˙3˙L5:
assumes A1: r –is total on″ G and A2: a∈G
shows a ≤ —a—
⟨*proof*⟩

$a^{-1} \leq |a|$ (in additive notation it would be $-a \leq |a|$.

lemma (in group3) OrderedGroup˙ZF˙3˙L6:
assumes A1: a∈G shows a^{-1} ≤ —a—
⟨*proof*⟩

Some inequalities about the product of two elements of a linearly ordered group and its absolute value.

lemma (in group3) OrderedGroup˙ZF˙3˙L6A:
assumes r –is total on″ G and a∈G b∈G
shows
a·b ≤—a—·—b—
a·b^{-1} ≤—a—·—b—
a^{-1}·b ≤—a—·—b—
a^{-1}·b^{-1} ≤—a—·—b—
⟨*proof*⟩

$|a^{-1}| \leq |a|$.

lemma (in group3) OrderedGroup˙ZF˙3˙L7:

assumes r –is total on″ G and a∈G
shows —a^{-1}—≤—a—
⟨*proof*⟩

$|a^{-1}| = |a|$.

lemma (in group3) OrderedGroup˙ZF˙3˙L7A:
assumes A1: r –is total on″ G and A2: a∈G
shows —a^{-1}— = —a—
⟨*proof*⟩

$|a \cdot b^{-1}| = |b \cdot a^{-1}|$. It doesn't look so strange in the additive notation: $|a - b| = |b - a|$.

lemma (in group3) OrderedGroup˙ZF˙3˙L7B:
assumes A1: r –is total on″ G and A2: a∈G b∈G
shows —a·b^{-1}— = —b·a^{-1}—
⟨*proof*⟩

Triangle inequality for linearly ordered abelian groups. It would be nice to drop commutativity or give an example that shows we can't do that.

theorem (in group3) OrdGroup˙triangle˙ineq:
assumes A1: P –is commutative on″ G
and A2: r –is total on″ G and A3: a∈G b∈G
shows —a·b— ≤ —a—·—b—
⟨*proof*⟩

We can multiply the sides of an inequality with absolute value.

lemma (in group3) OrderedGroup˙ZF˙3˙L7C:
assumes A1: P –is commutative on″ G
and A2: r –is total on″ G and A3: a∈G b∈G
and A4: —a— ≤ c —b— ≤ d
shows —a·b— ≤ c·d
⟨*proof*⟩

A version of the OrderedGroup˙ZF˙3˙L7C but with multiplying by the inverse.

lemma (in group3) OrderedGroup˙ZF˙3˙L7CA:
assumes P –is commutative on″ G
and r –is total on″ G and a∈G b∈G
and —a— ≤ c —b— ≤ d
shows —a·b^{-1}— ≤ c·d
⟨*proof*⟩

Triangle inequality with three integers.

lemma (in group3) OrdGroup˙triangle˙ineq3:
assumes A1: P –is commutative on″ G
and A2: r –is total on″ G and A3: a∈G b∈G c∈G
shows —a·b·c— ≤ —a—·—b—·—c—
⟨*proof*⟩

Some variants of the triangle inequality.

lemma (in group3) OrderedGroup˙ZF˙3˙L7D:
  assumes A1: P –is commutative on˝ G
  and A2: r –is total on˝ G and A3: a∈G b∈G
  and A4: —a·b^{-1}— ≤ c
  shows
  —a— ≤ c·—b—
  —a— ≤ —b—·c
  c^{-1}·a ≤ b
  a·c^{-1} ≤ b
  a ≤ b·c
⟨*proof*⟩

Some more variants of the triangle inequality.

lemma (in group3) OrderedGroup˙ZF˙3˙L7E:
  assumes A1: P –is commutative on˝ G
  and A2: r –is total on˝ G and A3: a∈G b∈G
  and A4: —a·b^{-1}— ≤ c
  shows b·c^{-1} ≤ a
⟨*proof*⟩

An application of the triangle inequality with four group elements.

lemma (in group3) OrderedGroup˙ZF˙3˙L7F:
  assumes A1: P –is commutative on˝ G
  and A2: r –is total on˝ G and
  A3: a∈G b∈G c∈G d∈G
  shows —a·c^{-1}— ≤ —a·b—·—c·d—·—b·d^{-1}—
⟨*proof*⟩

$|a| \leq L$ implies $L^{-1} \leq a$ (it would be $-L \leq a$ in the additive notation).

lemma (in group3) OrderedGroup˙ZF˙3˙L8:
  assumes A1: a∈G and A2: —a—≤L
   shows
  L^{-1}≤a
⟨*proof*⟩

In linearly ordered groups $|a| \leq L$ implies $a \leq L$ (it would be $a \leq L$ in the additive notation).

lemma (in group3) OrderedGroup˙ZF˙3˙L8A:
  assumes A1: r –is total on˝ G
  and A2: a∈G and A3: —a—≤L
  shows
  a≤L
  1≤L
⟨*proof*⟩

A somewhat generalized version of the above lemma.

lemma (in group3) OrderedGroup˙ZF˙3˙L8B:

assumes A1: a∈G and A2: —a—≤L and A3: **1**≤c
shows $(L\cdot c)^{-1}$ ≤ a
⟨*proof*⟩

If b is between a and $a \cdot c$, then $b \cdot a^{-1} \leq c$.

lemma (in group3) OrderedGroup˙ZF˙3˙L8C:
assumes A1: a≤b and A2: c∈G and A3: b≤c·a
shows —b·a^{-1}— ≤ c
⟨*proof*⟩

For linearly ordered groups if the absolute values of elements in a set are bounded, then the set is bounded.

lemma (in group3) OrderedGroup˙ZF˙3˙L9:
assumes A1: r –is total on˝ G
and A2: A⊆G and A3: ∀a∈A. —a— ≤ L
shows IsBounded(A,r)
⟨*proof*⟩

A slightly more general version of the previous lemma, stating the same fact for a set defined by separation.

lemma (in group3) OrderedGroup˙ZF˙3˙L9A:
assumes A1: r –is total on˝ G
and A2: ∀x∈X. b(x)∈G ∧ —b(x)—≤L
shows IsBounded(–b(x). x∈X˝,r)
⟨*proof*⟩

A special form of the previous lemma stating a similar fact for an image of a set by a function with values in a linearly ordered group.

lemma (in group3) OrderedGroup˙ZF˙3˙L9B:
assumes A1: r –is total on˝ G
and A2: f:X→G and A3: A⊆X
and A4: ∀x∈A. —f(x)— ≤ L
shows IsBounded(f(A),r)
⟨*proof*⟩

For linearly ordered groups if $l \leq a \leq u$ then $|a|$ is smaller than the greater of $|l|, |u|$.

lemma (in group3) OrderedGroup˙ZF˙3˙L10:
assumes A1: r –is total on˝ G
and A2: l≤a a≤u
shows
—a— ≤ GreaterOf(r,—l—,—u—)
⟨*proof*⟩

For linearly ordered groups if a set is bounded then the absolute values are bounded.

lemma (in group3) OrderedGroup˙ZF˙3˙L10A:

assumes A1: r –is total on″ G
and A2: IsBounded(A,r)
shows ∃L. ∀a∈A. —a— ≤ L
⟨*proof*⟩

A slightly more general version of the previous lemma, stating the same fact for a set defined by separation.

lemma (in group3) OrderedGroup˙ZF˙3˙L11:
assumes r –is total on″ G
and IsBounded(–b(x).x∈X″,r)
shows ∃L. ∀x∈X. —b(x)— ≤ L
⟨*proof*⟩

Absolute values of elements of a finite image of a nonempty set are bounded by an element of the group.

lemma (in group3) OrderedGroup˙ZF˙3˙L11A:
assumes A1: r –is total on″ G
and A2: X≠0 and A3: –b(x). x∈X″ ∈ Fin(G)
shows ∃L∈G. ∀x∈X. —b(x)— ≤ L
⟨*proof*⟩

In totally oredered groups the absolute value of a nonunit element is in G_+.

lemma (in group3) OrderedGroup˙ZF˙3˙L12:
assumes A1: r –is total on″ G
and A2: a∈G and A3: a≠**1**
shows —a— ∈ G_+
⟨*proof*⟩

34.2 Maximum absolute value of a set

Quite often when considering inequalities we prefer to talk about the absolute values instead of raw elements of a set. This section formalizes some material that is useful for that.

If a set has a maximum and minimum, then the greater of the absolute value of the maximum and minimum belongs to the image of the set by the absolute value function.

lemma (in group3) OrderedGroup˙ZF˙4˙L1:
assumes A ⊆ G
and HasAmaximum(r,A) HasAminimum(r,A)
and M = GreaterOf(r,—Minimum(r,A)—,—Maximum(r,A)—)
shows M ∈ AbsoluteValue(G,P,r)(A)
⟨*proof*⟩

If a set has a maximum and minimum, then the greater of the absolute value of the maximum and minimum bounds absolute values of all elements of the set.

lemma (in group3) OrderedGroup˙ZF˙4˙L2:
  assumes A1: r –is total on″ G
  and A2: HasAmaximum(r,A) HasAminimum(r,A)
  and A3: a∈A
  shows —a—≤ GreaterOf(r,—Minimum(r,A)—,—Maximum(r,A)—)
⟨*proof*⟩

If a set has a maximum and minimum, then the greater of the absolute value of the maximum and minimum bounds absolute values of all elements of the set. In this lemma the absolute values of ekements of a set are represented as the elements of the image of the set by the absolute value function.

lemma (in group3) OrderedGroup˙ZF˙4˙L3:
  assumes r –is total on″ G and A ⊆ G
  and HasAmaximum(r,A) HasAminimum(r,A)
  and b ∈ AbsoluteValue(G,P,r)(A)
  shows b≤ GreaterOf(r,—Minimum(r,A)—,—Maximum(r,A)—)
  ⟨*proof*⟩

If a set has a maximum and minimum, then the set of absolute values also has a maximum.

lemma (in group3) OrderedGroup˙ZF˙4˙L4:
  assumes A1: r –is total on″ G and A2: A ⊆ G
  and A3: HasAmaximum(r,A) HasAminimum(r,A)
  shows HasAmaximum(r,AbsoluteValue(G,P,r)(A))
⟨*proof*⟩

If a set has a maximum and a minimum, then all absolute values are bounded by the maximum of the set of absolute values.

lemma (in group3) OrderedGroup˙ZF˙4˙L5:
  assumes A1: r –is total on″ G and A2: A ⊆ G
  and A3: HasAmaximum(r,A) HasAminimum(r,A)
  and A4: a∈A
  shows —a— ≤ Maximum(r,AbsoluteValue(G,P,r)(A))
⟨*proof*⟩

34.3 Alternative definitions

Sometimes it is usful to define the order by prescibing the set of positive or nonnegative elements. This section deals with two such definitions. One takes a subset H of G that is closed under the group operation, $1 \notin H$ and for every $a \in H$ we have either $a \in H$ or $a^{-1} \in H$. Then the order is defined as $a \leq b$ iff $a = b$ or $a^{-1}b \in H$. For abelian groups this makes a linearly ordered group. We will refer to order defined this way in the comments as the order defined by a positive set. The context used in this section is the group0 context defined in Group˙ZF theory. Recall that f in that context denotes the group operation (unlike in the previous sections where the group operation was denoted P.

The order defined by a positive set is the same as the order defined by a nonnegative set.

lemma (in group0) OrderedGroup˙ZF˙5˙L1:
  assumes A1: r = –p ∈ G×G. fst(p) = snd(p) ∨ fst(p)$^{-1}$·snd(p) ∈ H″
  shows ⟨a,b⟩ ∈ r ⟷ a∈G ∧ b∈G ∧ a^{-1}·b ∈ H ∪ –**1**″
⟨*proof*⟩

The relation defined by a positive set is antisymmetric.

lemma (in group0) OrderedGroup˙ZF˙5˙L2:
  assumes A1: r = –p ∈ G×G. fst(p) = snd(p) ∨ fst(p)$^{-1}$·snd(p) ∈ H″
  and A2: ∀a∈G. a≠**1** ⟶ (a∈H) Xor (a^{-1}∈H)
  shows antisym(r)
⟨*proof*⟩

The relation defined by a positive set is transitive.

lemma (in group0) OrderedGroup˙ZF˙5˙L3:
  assumes A1: r = –p ∈ G×G. fst(p) = snd(p) ∨ fst(p)$^{-1}$·snd(p) ∈ H″
  and A2: H⊆G H –is closed under″ P
  shows trans(r)
⟨*proof*⟩

The relation defined by a positive set is translation invariant. With our definition this step requires the group to be abelian.

lemma (in group0) OrderedGroup˙ZF˙5˙L4:
  assumes A1: r = –p ∈ G×G. fst(p) = snd(p) ∨ fst(p)$^{-1}$·snd(p) ∈ H″
  and A2: P –is commutative on″ G
  and A3: ⟨a,b⟩ ∈ r and A4: c∈G
  shows ⟨a·c,b·c⟩ ∈ r ∧ ⟨c·a,c·b⟩ ∈ r
⟨*proof*⟩

If $H \subseteq G$ is closed under the group operation $1 \notin H$ and for every $a \in H$ we have either $a \in H$ or $a^{-1} \in H$, then the relation "$\leq$" defined by $a \leq b \Leftrightarrow a^{-1}b \in H$ orders the group G. In such order H may be the set of positive or nonnegative elements.

lemma (in group0) OrderedGroup˙ZF˙5˙L5:
  assumes A1: P –is commutative on″ G
  and A2: H⊆G H –is closed under″ P
  and A3: ∀a∈G. a≠**1** ⟶ (a∈H) Xor (a^{-1}∈H)
  and A4: r = –p ∈ G×G. fst(p) = snd(p) ∨ fst(p)$^{-1}$·snd(p) ∈ H″
  shows
  IsAnOrdGroup(G,P,r)
  r –is total on″ G
  Nonnegative(G,P,r) = PositiveSet(G,P,r) ∪ –**1**″
⟨*proof*⟩

If the set defined as in OrderedGroup˙ZF˙5˙L4 does not contain the neutral element, then it is the positive set for the resulting order.

lemma (in group0) OrderedGroup_ZF_5_L6:
assumes P {is commutative on} G
and H⊆G and $\mathbf{1}$ ∉ H
and r = {p ∈ G×G. fst(p) = snd(p) ∨ fst(p)$^{-1}$·snd(p) ∈ H}
shows PositiveSet(G,P,r) = H
⟨*proof*⟩

The next definition describes how we construct an order relation from the prescribed set of positive elements.

definition
OrderFromPosSet(G,P,H) ≡
{p ∈ G×G. fst(p) = snd(p) ∨ P⟨GroupInv(G,P)(fst(p)),snd(p)⟩ ∈ H }

The next theorem rephrases lemmas OrderedGroup_ZF_5_L5 and OrderedGroup_ZF_5_L6 using the definition of the order from the positive set OrderFromPosSet. To summarize, this is what it says: Suppose that $H \subseteq G$ is a set closed under that group operation such that $1 \notin H$ and for every nonunit group element a either $a \in H$ or $a^{-1} \in H$. Define the order as $a \leq b$ iff $a = b$ or $a^{-1} \cdot b \in H$. Then this order makes G into a linearly ordered group such H is the set of positive elements (and then of course $H \cup \{1\}$ is the set of nonnegative elements).

theorem (in group0) Group_ord_by_positive_set:
assumes P {is commutative on} G
and H⊆G H {is closed under} P $\mathbf{1}$ ∉ H
and ∀a∈G. a≠$\mathbf{1}$ ⟶ (a∈H) Xor (a^{-1}∈H)
shows
IsAnOrdGroup(G,P,OrderFromPosSet(G,P,H))
OrderFromPosSet(G,P,H) {is total on} G
PositiveSet(G,P,OrderFromPosSet(G,P,H)) = H
Nonnegative(G,P,OrderFromPosSet(G,P,H)) = H ∪ {$\mathbf{1}$}
⟨*proof*⟩

34.4 Odd Extensions

In this section we verify properties of odd extensions of functions defined on G_+. An odd extension of a function $f : G_+ \to G$ is a function $f^\circ : G \to G$ defined by $f^\circ(x) = f(x)$ if $x \in G_+$, $f(1) = 1$ and $f^\circ(x) = (f(x^{-1}))^{-1}$ for $x < 1$. Such function is the unique odd function that is equal to f when restricted to G_+.

The next lemma is just to see the definition of the odd extension in the notation used in the group1 context.

lemma (in group3) OrderedGroup_ZF_6_L1:
shows f° = f ∪ {⟨a, (f(a^{-1}))$^{-1}$⟩. a ∈ -G$_+$} ∪ {⟨$\mathbf{1}$,$\mathbf{1}$⟩}
⟨*proof*⟩

A technical lemma that states that from a function defined on G$_+$ with values in G we have $(f(a^{-1}))^{-1} \in G$.

lemma (in group3) OrderedGroup_ZF_6_L2:
  assumes f: $G_+ \to G$ and $a \in -G_+$
  shows
  $f(a^{-1}) \in G$
  $(f(a^{-1}))^{-1} \in G$
  ⟨*proof*⟩

The main theorem about odd extensions. It basically says that the odd extension of a function is what we want to to be.

lemma (in group3) odd_ext_props:
  assumes A1: r {is total on} G and A2: f: $G_+ \to G$
  shows
  $f^\circ : G \to G$
  $\forall a \in G_+. (f^\circ)(a) = f(a)$
  $\forall a \in (-G_+). (f^\circ)(a) = (f(a^{-1}))^{-1}$
  $(f^\circ)(\mathbf{1}) = \mathbf{1}$
⟨*proof*⟩

Odd extensions are odd, of course.

lemma (in group3) oddext_is_odd:
  assumes A1: r {is total on} G and A2: f: $G_+ \to G$
  and A3: $a \in G$
  shows $(f^\circ)(a^{-1}) = ((f^\circ)(a))^{-1}$
⟨*proof*⟩

Another way of saying that odd extensions are odd.

lemma (in group3) oddext_is_odd_alt:
  assumes A1: r {is total on} G and A2: f: $G_+ \to G$
  and A3: $a \in G$
  shows $((f^\circ)(a^{-1}))^{-1} = (f^\circ)(a)$
⟨*proof*⟩

34.5 Functions with infinite limits

In this section we consider functions $f : G \to G$ with the property that for $f(x)$ is arbitrarily large for large enough x. More precisely, for every $a \in G$ there exist $b \in G_+$ such that for every $x \geq b$ we have $f(x) \geq a$. In a sense this means that $\lim_{x\to\infty} f(x) = \infty$, hence the title of this section. We also prove dual statements for functions such that $\lim_{x\to-\infty} f(x) = -\infty$.

If an image of a set by a function with infinite positive limit is bounded above, then the set itself is bounded above.

lemma (in group3) OrderedGroup_ZF_7_L1:
  assumes A1: r {is total on} G and A2: $G \neq \{\mathbf{1}\}$ and
  A3: $f:G\to G$ and
  A4: $\forall a\in G.\exists b\in G_+.\forall x.\ b\leq x \longrightarrow a \leq f(x)$ and
  A5: $A\subseteq G$ and

A6: IsBoundedAbove(f(A),r)
shows IsBoundedAbove(A,r)
⟨*proof*⟩

If an image of a set defined by separation by a function with infinite positive limit is bounded above, then the set itself is bounded above.

lemma (in group3) OrderedGroup˙ZF˙7˙L2:
assumes A1: r –is total on″ G and A2: G $\neq$ –**1**″ and
A3: X$\neq$0 and A4: f:G$\rightarrow$G and
A5: $\forall$a$\in$G.$\exists$b$\in$$G_+$.$\forall$y. b$\leq$y $\longrightarrow$ a $\leq$ f(y) and
A6: $\forall$x$\in$X. b(x) $\in$ G $\wedge$ f(b(x)) $\leq$ U
shows $\exists$u.$\forall$x$\in$X. b(x) $\leq$ u
⟨*proof*⟩

If the image of a set defined by separation by a function with infinite negative limit is bounded below, then the set itself is bounded above. This is dual to OrderedGroup˙ZF˙7˙L2.

lemma (in group3) OrderedGroup˙ZF˙7˙L3:
assumes A1: r –is total on″ G and A2: G $\neq$ –**1**″ and
A3: X$\neq$0 and A4: f:G$\rightarrow$G and
A5: $\forall$a$\in$G.$\exists$b$\in$$G_+$.$\forall$y. b$\leq$y $\longrightarrow$ f(y^{-1}) $\leq$ a and
A6: $\forall$x$\in$X. b(x) $\in$ G $\wedge$ L $\leq$ f(b(x))
shows $\exists$l.$\forall$x$\in$X. l $\leq$ b(x)
⟨*proof*⟩

The next lemma combines OrderedGroup˙ZF˙7˙L2 and OrderedGroup˙ZF˙7˙L3 to show that if an image of a set defined by separation by a function with infinite limits is bounded, then the set itself i bounded.

lemma (in group3) OrderedGroup˙ZF˙7˙L4:
assumes A1: r –is total on″ G and A2: G $\neq$ –**1**″ and
A3: X$\neq$0 and A4: f:G$\rightarrow$G and
A5: $\forall$a$\in$G.$\exists$b$\in$$G_+$.$\forall$y. b$\leq$y $\longrightarrow$ a $\leq$ f(y) and
A6: $\forall$a$\in$G.$\exists$b$\in$$G_+$.$\forall$y. b$\leq$y $\longrightarrow$ f(y^{-1}) $\leq$ a and
A7: $\forall$x$\in$X. b(x) $\in$ G $\wedge$ L $\leq$ f(b(x)) $\wedge$ f(b(x)) $\leq$ U
shows $\exists$M.$\forall$x$\in$X. —b(x)— $\leq$ M
⟨*proof*⟩

end

35 Rings - introduction

theory Ring˙ZF imports AbelianGroup˙ZF

begin

This theory file covers basic facts about rings.

35.1 Definition and basic properties

In this section we define what is a ring and list the basic properties of rings.

We say that three sets (R, A, M) form a ring if (R, A) is an abelian group, (R, M) is a monoid and A is distributive with respect to M on R. A represents the additive operation on R. As such it is a subset of $(R \times R) \times R$ (recall that in ZF set theory functions are sets). Similarly M represents the multiplicative operation on R and is also a subset of $(R \times R) \times R$. We don't require the multiplicative operation to be commutative in the definition of a ring.

definition
IsAring(R,A,M) ≡ IsAgroup(R,A) ∧ (A –is commutative on″ R) ∧
IsAmonoid(R,M) ∧ IsDistributive(R,A,M)

We also define the notion of having no zero divisors. In standard notation the ring has no zero divisors if for all $a, b \in R$ we have $a \cdot b = 0$ implies $a = 0$ or $b = 0$.

definition
HasNoZeroDivs(R,A,M) ≡ (∀ a∈R. ∀ b∈R.
M⟨ a,b⟩ = TheNeutralElement(R,A) ⟶
a = TheNeutralElement(R,A) ∨ b = TheNeutralElement(R,A))

Next we define a locale that will be used when considering rings.

locale ring0 =

fixes R and A and M

assumes ringAssum: IsAring(R,A,M)

fixes ringa (infixl + 90)
defines ringa˙def [simp]: a+b ≡ A⟨ a,b⟩

fixes ringminus (- ˙ 89)
defines ringminus˙def [simp]: (-a) ≡ GroupInv(R,A)(a)

fixes ringsub (infixl - 90)
defines ringsub˙def [simp]: a-b ≡ a+(-b)

fixes ringm (infixl · 95)
defines ringm˙def [simp]: a·b ≡ M⟨ a,b⟩

fixes ringzero (**0**)
defines ringzero˙def [simp]: **0** ≡ TheNeutralElement(R,A)

fixes ringone (**1**)
defines ringone˙def [simp]: **1** ≡ TheNeutralElement(R,M)

fixes ringtwo (**2**)
defines ringtwo˙def [simp]: **2** ≡ **1**+**1**

fixes ringsq (ˆ2 [96] 97)
defines ringsq˙def [simp]: a^2 ≡ a·a

In the ring0 context we can use theorems proven in some other contexts.

lemma (in ring0) Ring˙ZF˙1˙L1: shows
monoid0(R,M)
group0(R,A)
A –is commutative on″ R
⟨*proof*⟩

The additive operation in a ring is distributive with respect to the multiplicative operation.

lemma (in ring0) ring˙oper˙distr: assumes A1: a∈R b∈R c∈R
shows
a·(b+c) = a·b + a·c
(b+c)·a = b·a + c·a
⟨*proof*⟩

Zero and one of the ring are elements of the ring. The negative of zero is zero.

lemma (in ring0) Ring˙ZF˙1˙L2:
shows **0**∈R **1**∈R (-**0**) = **0**
⟨*proof*⟩

The next lemma lists some properties of a ring that require one element of a ring.

lemma (in ring0) Ring˙ZF˙1˙L3: assumes a∈R
shows
(-a) ∈ R
(-(-a)) = a
a+**0** = a
0+a = a
a·**1** = a
1·a = a
a-a = **0**
a-**0** = a
2·a = a+a
(-a)+a = **0**
⟨*proof*⟩

Properties that require two elements of a ring.

lemma (in ring0) Ring˙ZF˙1˙L4: assumes A1: a∈R b∈R
shows
a+b ∈ R
a-b ∈ R

a·b ∈ R
a+b = b+a
⟨*proof*⟩

Cancellation of an element on both sides of equality. This is a property of groups, written in the (additive) notation we use for the additive operation in rings.

lemma (in ring0) ring_cancel_add:
assumes A1: a∈R b∈R and A2: a + b = a
shows b = **0**
⟨*proof*⟩

Any element of a ring multiplied by zero is zero.

lemma (in ring0) Ring_ZF_1_L6:
assumes A1: x∈R shows **0**·x = **0**   x·**0** = **0**
⟨*proof*⟩

Negative can be pulled out of a product.

lemma (in ring0) Ring_ZF_1_L7:
assumes A1: a∈R b∈R
shows
(-a)·b = -(a·b)
a·(-b) = -(a·b)
(-a)·b = a·(-b)
⟨*proof*⟩

Minus times minus is plus.

lemma (in ring0) Ring_ZF_1_L7A: assumes a∈R b∈R
shows (-a)·(-b) = a·b
⟨*proof*⟩

Subtraction is distributive with respect to multiplication.

lemma (in ring0) Ring_ZF_1_L8: assumes a∈R b∈R c∈R
shows
a·(b-c) = a·b - a·c
(b-c)·a = b·a - c·a
⟨*proof*⟩

Other basic properties involving two elements of a ring.

lemma (in ring0) Ring_ZF_1_L9: assumes a∈R b∈R
shows
(-b)-a = (-a)-b
(-(a+b)) = (-a)-b
(-(a-b)) = ((-a)+b)
a-(-b) = a+b
⟨*proof*⟩

If the difference of two element is zero, then those elements are equal.

lemma (in ring0) Ring_ZF_1_L9A:
  assumes A1: a∈R b∈R and A2: a-b = **0**
  shows a=b
⟨*proof*⟩

Other basic properties involving three elements of a ring.

lemma (in ring0) Ring_ZF_1_L10:
  assumes a∈R b∈R c∈R
  shows
  a+(b+c) = a+b+c

  a-(b+c) = a-b-c
  a-(b-c) = a-b+c
  ⟨*proof*⟩

Another property with three elements.

lemma (in ring0) Ring_ZF_1_L10A:
  assumes A1: a∈R b∈R c∈R
  shows a+(b-c) = a+b-c
  ⟨*proof*⟩

Associativity of addition and multiplication.

lemma (in ring0) Ring_ZF_1_L11:
  assumes a∈R b∈R c∈R
  shows
  a+b+c = a+(b+c)
  a·b·c = a·(b·c)
  ⟨*proof*⟩

An interpretation of what it means that a ring has no zero divisors.

lemma (in ring0) Ring_ZF_1_L12:
  assumes HasNoZeroDivs(R,A,M)
  and a∈R a≠**0** b∈R b≠**0**
  shows a·b≠**0**
  ⟨*proof*⟩

In rings with no zero divisors we can cancel nonzero factors.

lemma (in ring0) Ring_ZF_1_L12A:
  assumes A1: HasNoZeroDivs(R,A,M) and A2: a∈R b∈R c∈R
  and A3: a·c = b·c and A4: c≠**0**
  shows a=b
⟨*proof*⟩

In rings with no zero divisors if two elements are different, then after multiplying by a nonzero element they are still different.

lemma (in ring0) Ring_ZF_1_L12B:
  assumes A1: HasNoZeroDivs(R,A,M)
  a∈R b∈R c∈R a≠b c≠**0**

shows a·c ≠ b·c
⟨*proof*⟩

In rings with no zero divisors multiplying a nonzero element by a nonone element changes the value.

lemma (in ring0) Ring˙ZF˙1˙L12C:
assumes A1: HasNoZeroDivs(R,A,M) and
A2: a∈R b∈R and A3: **0**≠a **1**≠b
shows a ≠ a·b
⟨*proof*⟩

If a square is nonzero, then the element is nonzero.

lemma (in ring0) Ring˙ZF˙1˙L13:
assumes a∈R and $a^2 \neq \mathbf{0}$
shows a≠**0**
⟨*proof*⟩

Square of an element and its opposite are the same.

lemma (in ring0) Ring˙ZF˙1˙L14:
assumes a∈R shows $(-a)^2 = ((a)^2)$
⟨*proof*⟩

Adding zero to a set that is closed under addition results in a set that is also closed under addition. This is a property of groups.

lemma (in ring0) Ring˙ZF˙1˙L15:
assumes H ⊆ R and H –is closed under″ A
shows (H ∪ –**0**″) –is closed under″ A
⟨*proof*⟩

Adding zero to a set that is closed under multiplication results in a set that is also closed under multiplication.

lemma (in ring0) Ring˙ZF˙1˙L16:
assumes A1: H ⊆ R and A2: H –is closed under″ M
shows (H ∪ –**0**″) –is closed under″ M
⟨*proof*⟩

The ring is trivial iff $0 = 1$.

lemma (in ring0) Ring˙ZF˙1˙L17: shows R = –**0**″ ⟷ **0**=**1**
⟨*proof*⟩

The sets $\{m \cdot x.x \in R\}$ and $\{-m \cdot x.x \in R\}$ are the same.

lemma (in ring0) Ring˙ZF˙1˙L18: assumes A1: m∈R
shows –m·x. x∈R″ = –(-m)·x. x∈R″
⟨*proof*⟩

35.2 Rearrangement lemmas

In happens quite often that we want to show a fact like $(a+b)c+d = (ac+d-e)+(bc+e)$in rings. This is trivial in romantic math and probably there is a way to make it trivial in formalized math. However, I don't know any other way than to tediously prove each such rearrangement when it is needed. This section collects facts of this type.

Rearrangements with two elements of a ring.

lemma (in ring0) Ring˙ZF˙2˙L1: assumes a∈R b∈R
  shows a+b·a = (b+**1**)·a
  ⟨*proof*⟩

Rearrangements with two elements and cancelling.

lemma (in ring0) Ring˙ZF˙2˙L1A: assumes a∈R b∈R
  shows
  a-b+b = a
  a+b-a = b
  (-a)+b+a = b
  (-a)+(b+a) = b
  a+(b-a) = b
  ⟨*proof*⟩

In commutative rings $a-(b+1)c = (a-d-c)+(d-bc)$. For unknown reasons we have to use the raw set notation in the proof, otherwise all methods fail.

lemma (in ring0) Ring˙ZF˙2˙L2:
  assumes A1: a∈R b∈R c∈R d∈R
  shows a-(b+**1**)·c = (a-d-c)+(d-b·c)
⟨*proof*⟩

Rerrangement about adding linear functions.

lemma (in ring0) Ring˙ZF˙2˙L3:
  assumes A1: a∈R b∈R c∈R d∈R x∈R
  shows (a·x + b) + (c·x + d) = (a+c)·x + (b+d)
⟨*proof*⟩

Rearrangement with three elements

lemma (in ring0) Ring˙ZF˙2˙L4:
  assumes M –is commutative on˝ R
  and a∈R b∈R c∈R
  shows a·(b·c) = a·c·b
  ⟨*proof*⟩

Some other rearrangements with three elements.

lemma (in ring0) ring˙rearr˙3˙elemA:
  assumes A1: M –is commutative on˝ R and
  A2: a∈R b∈R c∈R

shows
a·(a·c) - b·(-b·c) = (a·a + b·b)·c
a·(-b·c) + b·(a·c) = **0**
⟨*proof*⟩

Some rearrangements with four elements. Properties of abelian groups.

lemma (in ring0) Ring_ZF_2_L5:
assumes a∈R b∈R c∈R d∈R
shows
a - b - c - d = a - d - b - c
a + b + c - d = a - d + b + c
a + b - c - d = a - c + (b - d)
a + b + c + d = a + c + (b + d)
⟨*proof*⟩

Two big rearranegements with six elements, useful for proving properties of complex addition and multiplication.

lemma (in ring0) Ring_ZF_2_L6:
assumes A1: a∈R b∈R c∈R d∈R e∈R f∈R
shows
a·(c·e - d·f) - b·(c·f + d·e) =
(a·c - b·d)·e - (a·d + b·c)·f
a·(c·f + d·e) + b·(c·e - d·f) =
(a·c - b·d)·f + (a·d + b·c)·e
a·(c+e) - b·(d+f) = a·c - b·d + (a·e - b·f)
a·(d+f) + b·(c+e) = a·d + b·c + (a·f + b·e)
⟨*proof*⟩

end

36 More on rings

theory Ring_ZF_1 imports Ring_ZF Group_ZF_3

begin

This theory is devoted to the part of ring theory specific the construction of real numbers in the Real_ZF_x series of theories. The goal is to show that classes of almost homomorphisms form a ring.

36.1 The ring of classes of almost homomorphisms

Almost homomorphisms do not form a ring as the regular homomorphisms do because the lifted group operation is not distributive with respect to composition – we have $s \circ (r \cdot q) \neq s \circ r \cdot s \circ q$ in general. However, we do have $s \circ (r \cdot q) \approx s \circ r \cdot s \circ q$ in the sense of the equivalence relation defined by the group of finite range functions (that is a normal subgroup of almost homomorphisms, if the group is abelian). This allows to define a natural ring structure on the classes of almost homomorphisms.

The next lemma provides a formula useful for proving that two sides of the distributive law equation for almost homomorphisms are almost equal.

lemma (in group1) Ring˙ZF˙1˙1˙L1:
  assumes A1: s∈AH r∈AH q∈AH and A2: n∈G
  shows
  $((s\circ(r\cdot q))(n))\cdot(((s\circ r)\cdot(s\circ q))(n))^{-1}= \delta(s,\langle r(n),q(n)\rangle)$
  $((r\cdot q)\circ s)(n) = ((r\circ s)\cdot(q\circ s))(n)$
⟨*proof*⟩

The sides of the distributive law equations for almost homomorphisms are almost equal.

lemma (in group1) Ring˙ZF˙1˙1˙L2:
  assumes A1: s∈AH r∈AH q∈AH
  shows
  $s\circ(r\cdot q) \approx (s\circ r)\cdot(s\circ q)$
  $(r\cdot q)\circ s = (r\circ s)\cdot(q\circ s)$
⟨*proof*⟩

The essential condition to show the distributivity for the operations defined on classes of almost homomorphisms.

lemma (in group1) Ring˙ZF˙1˙1˙L3:
  assumes A1: R = QuotientGroupRel(AH,Op1,FR)
  and A2: a ∈ AH//R b ∈ AH//R c ∈ AH//R
  and A3: A = ProjFun2(AH,R,Op1) M = ProjFun2(AH,R,Op2)
  shows $M\langle a,A\langle b,c\rangle\rangle = A\langle M\langle a,b\rangle,M\langle a,c\rangle\rangle \wedge$
  $M\langle A\langle b,c\rangle,a\rangle = A\langle M\langle b,a\rangle,M\langle c,a\rangle\rangle$
⟨*proof*⟩

The projection of the first group operation on almost homomorphisms is distributive with respect to the second group operation.

lemma (in group1) Ring˙ZF˙1˙1˙L4:
  assumes A1: R = QuotientGroupRel(AH,Op1,FR)
  and A2: A = ProjFun2(AH,R,Op1) M = ProjFun2(AH,R,Op2)
  shows IsDistributive(AH//R,A,M)
⟨*proof*⟩

The classes of almost homomorphisms form a ring.

theorem (in group1) Ring˙ZF˙1˙1˙T1:
  assumes R = QuotientGroupRel(AH,Op1,FR)
  and A = ProjFun2(AH,R,Op1) M = ProjFun2(AH,R,Op2)
  shows IsAring(AH//R,A,M)
  ⟨*proof*⟩

end

37 Ordered rings

theory OrderedRing˙ZF imports Ring˙ZF OrderedGroup˙ZF˙1

begin

In this theory file we consider ordered rings.

37.1 Definition and notation

This section defines ordered rings and sets up appriopriate notation.

We define ordered ring as a commutative ring with linear order that is preserved by translations and such that the set of nonnegative elements is closed under multiplication. Note that this definition does not guarantee that there are no zero divisors in the ring.

definition
IsAnOrdRing(R,A,M,r) ≡
(IsAring(R,A,M) ∧ (M –is commutative on˝ R) ∧
r⊆R×R ∧ IsLinOrder(R,r) ∧
(∀a b. ∀ c∈R. ⟨ a,b⟩ ∈ r ⟶ ⟨A⟨ a,c⟩,A⟨ b,c⟩⟩ ∈ r) ∧
(Nonnegative(R,A,r) –is closed under˝ M))

The next context (locale) defines notation used for ordered rings. We do that by extending the notation defined in the ring0 locale and adding some assumptions to make sure we are talking about ordered rings in this context.

locale ring1 = ring0 +

assumes mult˙commut: M –is commutative on˝ R

fixes r

assumes ordincl: r ⊆ R×R

assumes linord: IsLinOrder(R,r)

fixes lesseq (infix ≤ 68)
defines lesseq˙def [simp]: a ≤ b ≡ ⟨ a,b⟩ ∈ r

fixes sless (infix ¡ 68)
defines sless˙def [simp]: a ¡ b ≡ a≤b ∧ a≠b

assumes ordgroup: ∀a b. ∀ c∈R. a≤b ⟶ a+c ≤ b+c

assumes pos˙mult˙closed: Nonnegative(R,A,r) –is closed under˝ M

fixes abs (— ˙ —)
defines abs˙def [simp]: —a— ≡ AbsoluteValue(R,A,r)(a)

fixes positiveset (R_+)
defines positiveset˙def [simp]: $R_+ \equiv$ PositiveSet(R,A,r)

The next lemma assures us that we are talking about ordered rings in the ring1 context.

lemma (in ring1) OrdRing˙ZF˙1˙L1: shows IsAnOrdRing(R,A,M,r)
⟨*proof*⟩

We can use theorems proven in the ring1 context whenever we talk about an ordered ring.

lemma OrdRing˙ZF˙1˙L2: assumes IsAnOrdRing(R,A,M,r)
shows ring1(R,A,M,r)
⟨*proof*⟩

In the ring1 context $a \leq b$ implies that a, b are elements of the ring.

lemma (in ring1) OrdRing˙ZF˙1˙L3: assumes a≤b
shows a∈R b∈R
⟨*proof*⟩

Ordered ring is an ordered group, hence we can use theorems proven in the group3 context.

lemma (in ring1) OrdRing˙ZF˙1˙L4: shows
IsAnOrdGroup(R,A,r)
r –is total on˝ R
A –is commutative on˝ R
group3(R,A,r)
⟨*proof*⟩

The order relation in rings is transitive.

lemma (in ring1) ring˙ord˙transitive: assumes A1: a≤b b≤c
shows a≤c
⟨*proof*⟩

Transitivity for the strict order: if $a < b$ and $b \leq c$, then $a < c$. Property of ordered groups.

lemma (in ring1) ring˙strict˙ord˙trans:
assumes A1: a¡b and A2: b≤c
shows a¡c
⟨*proof*⟩

Another version of transitivity for the strict order: if $a \leq b$ and $b < c$, then $a < c$. Property of ordered groups.

lemma (in ring1) ring˙strict˙ord˙transit:
assumes A1: a≤b and A2: b¡c
shows a¡c
⟨*proof*⟩

The next lemma shows what happens when one element of an ordered ring is not greater or equal than another.

lemma (in ring1) OrdRing˙ZF˙1˙L4A: assumes A1: a∈R b∈R
and A2: ¬(a≤b)
shows b ≤ a (-a) ≤ (-b) a≠b
⟨*proof*⟩

A special case of OrdRing˙ZF˙1˙L4A when one of the constants is 0. This is useful for many proofs by cases.

corollary (in ring1) ord˙ring˙split2: assumes A1: a∈R
shows a≤**0** ∨ (**0**≤a ∧ a≠**0**)
⟨*proof*⟩

Taking minus on both sides reverses an inequality.

lemma (in ring1) OrdRing˙ZF˙1˙L4B: assumes a≤b
shows (-b) ≤ (-a)
⟨*proof*⟩

The next lemma just expands the condition that requires the set of non-negative elements to be closed with respect to multiplication. These are properties of totally ordered groups.

lemma (in ring1) OrdRing˙ZF˙1˙L5:
assumes **0**≤a **0**≤b
shows **0** ≤ a·b
⟨*proof*⟩

Double nonnegative is nonnegative.

lemma (in ring1) OrdRing˙ZF˙1˙L5A: assumes A1: **0**≤a
shows **0**≤**2**·a
⟨*proof*⟩

A sufficient (somewhat redundant) condition for a structure to be an ordered ring. It says that a commutative ring that is a totally ordered group with respect to the additive operation such that set of nonnegative elements is closed under multiplication, is an ordered ring.

lemma OrdRing˙ZF˙1˙L6:
assumes
IsAring(R,A,M)
M –is commutative on″ R
Nonnegative(R,A,r) –is closed under″ M
IsAnOrdGroup(R,A,r)
r –is total on″ R
shows IsAnOrdRing(R,A,M,r)
⟨*proof*⟩

$a \leq b$ iff $a - b \leq 0$. This is a fact from OrderedGroup.thy, where it is stated in multiplicative notation.

lemma (in ring1) OrdRing_ZF_1_L7:
  assumes a∈R b∈R
  shows a≤b ⟷ a-b ≤ **0**
  ⟨*proof*⟩

Negative times positive is negative.

lemma (in ring1) OrdRing_ZF_1_L8:
  assumes A1: a≤**0** and A2: **0**≤b
  shows a·b ≤ **0**
⟨*proof*⟩

We can multiply both sides of an inequality by a nonnegative ring element. This property is sometimes (not here) used to define ordered rings.

lemma (in ring1) OrdRing_ZF_1_L9:
  assumes A1: a≤b and A2: **0**≤c
  shows
  a·c ≤ b·c
  c·a ≤ c·b
⟨*proof*⟩

A special case of OrdRing_ZF_1_L9: we can multiply an inequality by a positive ring element.

lemma (in ring1) OrdRing_ZF_1_L9A:
  assumes A1: a≤b and A2: c∈R_+
  shows
  a·c ≤ b·c
  c·a ≤ c·b
⟨*proof*⟩

A square is nonnegative.

lemma (in ring1) OrdRing_ZF_1_L10:
  assumes A1: a∈R shows **0**≤(a^2)
⟨*proof*⟩

1 is nonnegative.

corollary (in ring1) ordring_one_is_nonneg: shows **0** ≤ **1**
⟨*proof*⟩

In nontrivial rings one is positive.

lemma (in ring1) ordring_one_is_pos: assumes **0**≠**1**
  shows **1** ∈ R_+
  ⟨*proof*⟩

Nonnegative is not negative. Property of ordered groups.

lemma (in ring1) OrdRing_ZF_1_L11: assumes **0**≤a
  shows ¬(a≤**0** ∧ a≠**0**)
  ⟨*proof*⟩

A negative element cannot be a square.

lemma (in ring1) OrdRing˙ZF˙1˙L12:
  assumes A1: a≤**0** a≠**0**
  shows ¬(∃ b∈R. a = (b^2))
⟨*proof*⟩

If $a \leq b$, then $0 \leq b - a$.

lemma (in ring1) OrdRing˙ZF˙1˙L13: assumes a≤b
  shows **0** ≤ b-a
  ⟨*proof*⟩

If $a < b$, then $0 < b - a$.

lemma (in ring1) OrdRing˙ZF˙1˙L14: assumes a≤b a≠b
  shows
  0 ≤ b-a **0** ≠ b-a
  b-a ∈ R_+
  ⟨*proof*⟩

If the difference is nonnegative, then $a \leq b$.

lemma (in ring1) OrdRing˙ZF˙1˙L15:
  assumes a∈R b∈R and **0** ≤ b-a
  shows a≤b
  ⟨*proof*⟩

A nonnegative number is does not decrease when multiplied by a number greater or equal 1.

lemma (in ring1) OrdRing˙ZF˙1˙L16:
  assumes A1: **0**≤a and A2: **1**≤b
  shows a≤a·b
⟨*proof*⟩

We can multiply the right hand side of an inequality between nonnegative ring elements by an element greater or equal 1.

lemma (in ring1) OrdRing˙ZF˙1˙L17:
  assumes A1: **0**≤a and A2: a≤b and A3: **1**≤c
  shows a≤b·c
⟨*proof*⟩

Strict order is preserved by translations.

lemma (in ring1) ring˙strict˙ord˙trans˙inv:
  assumes a¡b and c∈R
  shows
  a+c ¡ b+c
  c+a ¡ c+b
  ⟨*proof*⟩

We can put an element on the other side of a strict inequality, changing its sign.

lemma (in ring1) OrdRing˙ZF˙1˙L18:
assumes a∈R b∈R and a-b ¡ c
shows a ¡ c+b
⟨*proof*⟩

We can add the sides of two inequalities, the first of them strict, and we get a strict inequality. Property of ordered groups.

lemma (in ring1) OrdRing˙ZF˙1˙L19:
assumes a¡b and c≤d
shows a+c ¡ b+d
⟨*proof*⟩

We can add the sides of two inequalities, the second of them strict and we get a strict inequality. Property of ordered groups.

lemma (in ring1) OrdRing˙ZF˙1˙L20:
assumes a≤b and c¡d
shows a+c ¡ b+d
⟨*proof*⟩

37.2 Absolute value for ordered rings

Absolute value is defined for ordered groups as a function that is the identity on the nonnegative set and the negative of the element (the inverse in the multiplicative notation) on the rest. In this section we consider properties of absolute value related to multiplication in ordered rings.

Absolute value of a product is the product of absolute values: the case when both elements of the ring are nonnegative.

lemma (in ring1) OrdRing˙ZF˙2˙L1:
assumes **0**≤a **0**≤b
shows —a·b— = —a—·—b—
⟨*proof*⟩

The absolue value of an element and its negative are the same.

lemma (in ring1) OrdRing˙ZF˙2˙L2: assumes a∈R
shows —-a— = —a—
⟨*proof*⟩

The next lemma states that $|a \cdot (-b)| = |(-a) \cdot b| = |(-a) \cdot (-b)| = |a \cdot b|$.

lemma (in ring1) OrdRing˙ZF˙2˙L3:
assumes a∈R b∈R
shows
—(-a)·b— = —a·b—
—a·(-b)— = —a·b—
—(-a)·(-b)— = —a·b—
⟨*proof*⟩

This lemma allows to prove theorems for the case of positive and negative elements of the ring separately.

lemma (in ring1) OrdRing˙ZF˙2˙L4: assumes a∈R and ¬(**0**≤a)
  shows **0** ≤ (-a) **0**≠a
  ⟨*proof*⟩

Absolute value of a product is the product of absolute values.

lemma (in ring1) OrdRing˙ZF˙2˙L5:
  assumes A1: a∈R b∈R
  shows —a·b— = —a—·—b—
⟨*proof*⟩

Triangle inequality. Property of linearly ordered abelian groups.

lemma (in ring1) ord˙ring˙triangle˙ineq: assumes a∈R b∈R
  shows —a+b— ≤ —a—+—b—
  ⟨*proof*⟩

If $a \leq c$ and $b \leq c$, then $a + b \leq 2 \cdot c$.

lemma (in ring1) OrdRing˙ZF˙2˙L6:
  assumes a≤c b≤c shows a+b ≤ **2**·c
  ⟨*proof*⟩

37.3 Positivity in ordered rings

This section is about properties of the set of positive elements R_+.

The set of positive elements is closed under ring addition. This is a property of ordered groups, we just reference a theorem from OrderedGroup˙ZF theory in the proof.

lemma (in ring1) OrdRing˙ZF˙3˙L1: shows R_+ –is closed under″ A
  ⟨*proof*⟩

Every element of a ring can be either in the postitive set, equal to zero or its opposite (the additive inverse) is in the positive set. This is a property of ordered groups, we just reference a theorem from OrderedGroup˙ZF theory.

lemma (in ring1) OrdRing˙ZF˙3˙L2: assumes a∈R
  shows Exactly˙1˙of˙3˙holds (a=**0**, a∈R_+, (-a) ∈ R_+)
  ⟨*proof*⟩

If a ring element $a \neq 0$, and it is not positive, then $-a$ is positive.

lemma (in ring1) OrdRing˙ZF˙3˙L2A: assumes a∈R a≠**0** a ∉ R_+
  shows (-a) ∈ R_+
  ⟨*proof*⟩

R_+ is closed under multiplication iff the ring has no zero divisors.

lemma (in ring1) OrdRing˙ZF˙3˙L3:

shows (R_+ –is closed under″ M)⟷ HasNoZeroDivs(R,A,M)
⟨*proof*⟩

Another (in addition to OrdRing˙ZF˙1˙L6 sufficient condition that defines order in an ordered ring starting from the positive set.

theorem (in ring0) ring˙ord˙by˙positive˙set:
assumes
A1: M –is commutative on″ R and
A2: P⊆R P –is closed under″ A **0** ∉ P and
A3: ∀a∈R. a≠**0** ⟶ (a∈P) Xor ((-a) ∈ P) and
A4: P –is closed under″ M and
A5: r = OrderFromPosSet(R,A,P)
shows
IsAnOrdGroup(R,A,r)
IsAnOrdRing(R,A,M,r)
r –is total on″ R
PositiveSet(R,A,r) = P
Nonnegative(R,A,r) = P ∪ –**0**″
HasNoZeroDivs(R,A,M)
⟨*proof*⟩

Nontrivial ordered rings are infinite. More precisely we assume that the neutral element of the additive operation is not equal to the multiplicative neutral element and show that the the set of positive elements of the ring is not a finite subset of the ring and the ring is not a finite subset of itself.

theorem (in ring1) ord˙ring˙infinite: assumes **0**≠**1**
shows
R_+ ∉ Fin(R)
R ∉ Fin(R)
⟨*proof*⟩

If every element of a nontrivial ordered ring can be dominated by an element from B, then we B is not bounded and not finite.

lemma (in ring1) OrdRing˙ZF˙3˙L4:
assumes **0**≠**1** and ∀a∈R. ∃b∈B. a≤b
shows
¬IsBoundedAbove(B,r)
B ∉ Fin(R)
⟨*proof*⟩

If m is greater or equal the multiplicative unit, then the set $\{m \cdot n : n \in R\}$ is infinite (unless the ring is trivial).

lemma (in ring1) OrdRing˙ZF˙3˙L5: assumes A1: **0**≠**1** and A2: **1**≤m
shows
–m·x. x∈R_+″ ∉ Fin(R)
–m·x. x∈R″ ∉ Fin(R)
–(-m)·x. x∈R″ ∉ Fin(R)
⟨*proof*⟩

If m is less or equal than the negative of multiplicative unit, then the set $\{m \cdot n : n \in R\}$ is infinite (unless the ring is trivial).

lemma (in ring1) OrdRing˙ZF˙3˙L6: assumes A1: **0**≠**1** and A2: m ≤ **-1**
 shows –m·x. x∈R″ ∉ Fin(R)
⟨*proof*⟩

All elements greater or equal than an element of R_+ belong to R_+. Property of ordered groups.

lemma (in ring1) OrdRing˙ZF˙3˙L7: assumes A1: a ∈ R_+ and A2: a≤b
 shows b ∈ R_+
⟨*proof*⟩

A special case of OrdRing˙ZF˙3˙L7: a ring element greater or equal than 1 is positive.

corollary (in ring1) OrdRing˙ZF˙3˙L8: assumes A1: **0**≠**1** and A2: **1**≤a
 shows a ∈ R_+
⟨*proof*⟩

Adding a positive element to a strictly increases a. Property of ordered groups.

lemma (in ring1) OrdRing˙ZF˙3˙L9: assumes A1: a∈R b∈R_+
 shows a ≤ a+b a ≠ a+b
 ⟨*proof*⟩

A special case of OrdRing˙ZF˙3˙L9: in nontrivial rings adding one to a increases a.

corollary (in ring1) OrdRing˙ZF˙3˙L10: assumes A1: **0**≠**1** and A2: a∈R
 shows a ≤ a+**1** a ≠ a+**1**
 ⟨*proof*⟩

If a is not greater than b, then it is strictly less than $b + 1$.

lemma (in ring1) OrdRing˙ZF˙3˙L11: assumes A1: **0**≠**1** and A2: a≤b
 shows a¡ b+**1**
⟨*proof*⟩

For any ring element a the greater of a and 1 is a positive element that is greater or equal than m. If we add 1 to it we get a positive element that is strictly greater than m. This holds in nontrivial rings.

lemma (in ring1) OrdRing˙ZF˙3˙L12: assumes A1: **0**≠**1** and A2: a∈R
 shows
 a ≤ GreaterOf(r,**1**,a)
 GreaterOf(r,**1**,a) ∈ R_+
 GreaterOf(r,**1**,a) + **1** ∈ R_+
 a ≤ GreaterOf(r,**1**,a) + **1** a ≠ GreaterOf(r,**1**,a) + **1**
⟨*proof*⟩

We can multiply strict inequality by a positive element.

lemma (in ring1) OrdRing˙ZF˙3˙L13:
  assumes A1: HasNoZeroDivs(R,A,M) and
  A2: a¡b and A3: c∈R_+
  shows
  a·c ¡ b·c
  c·a ¡ c·b
⟨*proof*⟩

A sufficient condition for an element to be in the set of positive ring elements.

lemma (in ring1) OrdRing˙ZF˙3˙L14: assumes **0**≤a and a≠**0**
  shows a ∈ R_+
  ⟨*proof*⟩

If a ring has no zero divisors, the square of a nonzero element is positive.

lemma (in ring1) OrdRing˙ZF˙3˙L15:
  assumes HasNoZeroDivs(R,A,M) and a∈R a≠**0**
  shows **0** ≤ a^2 a^2 ≠ **0** a^2 ∈ R_+
  ⟨*proof*⟩

In rings with no zero divisors we can (strictly) increase a positive element by multiplying it by an element that is greater than 1.

lemma (in ring1) OrdRing˙ZF˙3˙L16:
  assumes HasNoZeroDivs(R,A,M) and a ∈ R_+ and **1**≤b **1**≠b
  shows a≤a·b a ≠ a·b
  ⟨*proof*⟩

If the right hand side of an inequality is positive we can multiply it by a number that is greater than one.

lemma (in ring1) OrdRing˙ZF˙3˙L17:
   assumes A1: HasNoZeroDivs(R,A,M) and A2: b∈R_+ and
  A3: a≤b and A4: **1**¡c
  shows a¡b·c
⟨*proof*⟩

We can multiply a right hand side of an inequality between positive numbers by a number that is greater than one.

lemma (in ring1) OrdRing˙ZF˙3˙L18:
  assumes A1: HasNoZeroDivs(R,A,M) and A2: a ∈ R_+ and
  A3: a≤b and A4: **1**¡c
  shows a¡b·c
⟨*proof*⟩

In ordered rings with no zero divisors if at least one of a, b is not zero, then $0 < a^2 + b^2$, in particular $a^2 + b^2 \neq 0$.

lemma (in ring1) OrdRing˙ZF˙3˙L19:
  assumes A1: HasNoZeroDivs(R,A,M) and A2: a∈R b∈R and
  A3: a ≠ **0** ∨ b ≠ **0**

shows **0** ¡ a^2 + b^2
⟨*proof*⟩

end

38 Cardinal numbers

theory Cardinal˙ZF imports ZF.CardinalArith func1

begin

This theory file deals with results on cardinal numbers (cardinals). Cardinals are a genaralization of the natural numbers, used to measure the cardinality (size) of sets. Contributed by Daniel de la Concepcion.

38.1 Some new ideas on cardinals

All the results of this section are done without assuming the Axiom of Choice. With the Axiom of Choice in play, the proofs become easier and some of the assumptions may be dropped.
Since General Topology Theory is closely related to Set Theory, it is very interesting to make use of all the possibilities of Set Theory to try to classify homeomorphic topological spaces. These ideas are generally used to prove that two topological spaces are not homeomorphic.

There exist cardinals which are the successor of another cardinal, but; as happens with ordinals, there are cardinals which are limit cardinal.

definition
LimitC(i) ≡ Card(i) ∧ 0¡i ∧ (∀ y. (y¡i∧Card(y)) ⟶ csucc(y)¡i)

Simple fact used a couple of times in proofs.

lemma nat˙less˙infty: assumes n∈nat and InfCard(X) shows n¡X
⟨*proof*⟩

There are three types of cardinals, the zero one, the succesors of other cardinals and the limit cardinals.

lemma Card˙cases˙disj:
assumes Card(i)
shows i=0 — (∃ j. Card(j) ∧ i=csucc(j)) — LimitC(i)
⟨*proof*⟩

Given an ordinal bounded by a cardinal in ordinal order, we can change to the order of sets.

lemma le˙imp˙lesspoll:
  assumes Card(Q)
  shows A $\leq$ Q $\Longrightarrow$ A $\lesssim$ Q
$\langle proof \rangle$

There are two types of infinite cardinals, the natural numbers and those that have at least one infinite strictly smaller cardinal.

lemma InfCard˙cases˙disj:
  assumes InfCard(Q)
  shows Q=nat $\vee$ ($\exists$j. csucc(j)$\lesssim$Q $\wedge$ InfCard(j))
$\langle proof \rangle$

A more readable version of standard Isabelle/ZF Ord˙linear˙lt

lemma Ord˙linear˙lt˙IML: assumes Ord(i) Ord(j)
  shows i¡j $\vee$ i=j $\vee$ j¡i
  $\langle proof \rangle$

A set is injective and not bijective to the successor of a cardinal if and only if it is injective and possibly bijective to the cardinal.

lemma Card˙less˙csucc˙eq˙le:
  assumes Card(m)
  shows A $\prec$ csucc(m) $\longleftrightarrow$ A $\lesssim$ m
$\langle proof \rangle$

If the successor of a cardinal is infinite, so is the original cardinal.

lemma csucc˙inf˙imp˙inf:
  assumes Card(j) and InfCard(csucc(j))
  shows InfCard(j)
$\langle proof \rangle$

Since all the cardinals previous to nat are finite, it cannot be a successor cardinal; hence it is a LimitC cardinal.

corollary LimitC˙nat:
  shows LimitC(nat)
$\langle proof \rangle$

38.2 Main result on cardinals (without the Axiom of Choice)

If two sets are strictly injective to an infinite cardinal, then so is its union. For the case of successor cardinal, this theorem is done in the isabelle library in a more general setting; but that theorem is of not use in the case where LimitC(Q) and it also makes use of the Axiom of Choice. The mentioned theorem is in the theory file Cardinal˙AC.thy

Note that if Q is finite and different from 1, let's assume $Q = n$, then the union of A and B is not bounded by Q. Counterexample: two disjoint sets

of $n-1$ elements each have a union of $2n-2$ elements which are more than n.
Note also that if $Q=1$ then A and B must be empty and the union is then empty too; and Q cannot be 0 because no set is injective and not bijective to 0.
The proof is divided in two parts, first the case when both sets A and B are finite; and second, the part when at least one of them is infinite. In the first part, it is used the fact that a finite union of finite sets is finite. In the second part it is used the linear order on cardinals (ordinals). This proof can not be generalized to a setting with an infinite union easily.

lemma less˙less˙imp˙un˙less:
  assumes A≺Q and B≺Q and InfCard(Q)
  shows A ∪ B≺Q
⟨*proof*⟩

38.3 Choice axioms

We want to prove some theorems assuming that some version of the Axiom of Choice holds. To avoid introducing it as an axiom we will defin an appropriate predicate and put that in the assumptions of the theorems. That way technically we stay inside ZF.

The first predicate we define states that the axiom of Q-choice holds for subsets of K if we can find a choice function for every family of subsets of K whose (that family's) cardinality does not exceed Q.

definition
  AxiomCardinalChoice (–the axiom of″˙–choice holds for subsets″˙) where
  –the axiom of″ Q –choice holds for subsets″K ≡ Card(Q) ∧ (∀ M N. (M ≲Q ∧ (∀t∈M. Nt≠0 ∧ Nt⊆K)) ⟶ (∃f. f:Pi(M,λt. Nt) ∧ (∀t∈M. ft∈Nt)))

Next we define a general form of Q choice where we don't require a collection of files to be included in a file.

definition
  AxiomCardinalChoiceGen (–the axiom of″˙–choice holds″) where
  –the axiom of″ Q –choice holds″ ≡ Card(Q) ∧ (∀ M N. (M ≲Q ∧ (∀t∈M. Nt≠0)) ⟶ (∃f. f:Pi(M,λt. Nt) ∧ (∀t∈M. ft∈Nt)))

The axiom of finite choice always holds.

theorem finite˙choice:
  assumes n∈nat
  shows –the axiom of″ n –choice holds″
⟨*proof*⟩

The axiom of choice holds if and only if the AxiomCardinalChoice holds for every couple of a cardinal Q and a set K.

lemma choice˙subset˙imp˙choice:
shows –the axiom of" Q –choice holds" $\longleftrightarrow$ ($\forall$ K. –the axiom of" Q –choice holds for subsets"K)
⟨*proof*⟩

A choice axiom for greater cardinality implies one for smaller cardinality

lemma greater˙choice˙imp˙smaller˙choice:
assumes Q$\lesssim$Q1 Card(Q)
shows –the axiom of" Q1 –choice holds" $\longrightarrow$ (–the axiom of" Q –choice holds")
⟨*proof*⟩

If we have a surjective function from a set which is injective to a set of ordinals, then we can find an injection which goes the other way.

lemma surj˙fun˙inv:
assumes f $\in$ surj(A,B) A$\subseteq$Q Ord(Q)
shows B$\lesssim$A
⟨*proof*⟩

The difference with the previous result is that in this one A is not a subset of an ordinal, it is only injective with one.

theorem surj˙fun˙inv˙2:
assumes f:surj(A,B) A$\lesssim$Q Ord(Q)
shows B$\lesssim$A
⟨*proof*⟩

end

39 Groups 4

theory Group˙ZF˙4 imports Group˙ZF˙1 Group˙ZF˙2 Finite˙ZF Ring˙ZF
Cardinal˙ZF Semigroup˙ZF

begin

This theory file deals with normal subgroup test and some finite group theory. Then we define group homomorphisms and prove that the set of endomorphisms forms a ring with unity and we also prove the first isomorphism theorem.

39.1 Conjugation of subgroups

The conjugate of a subgroup is a subgroup.

theorem(in group0) semigr0:
shows semigr0(G,P)
⟨*proof*⟩

theorem (in group0) conj˙group˙is˙group:
assumes IsAsubgroup(H,P) g∈G
shows IsAsubgroup(–g·(h·g^{-1}). h∈H″,P)
⟨*proof*⟩

Every set is equipollent with its conjugates.

theorem (in group0) conj˙set˙is˙eqpoll:
assumes H⊆G g∈G
shows H≈–g·(h·g^{-1}). h∈H″
⟨*proof*⟩

Every normal subgroup contains its conjugate subgroups.

theorem (in group0) norm˙group˙cont˙conj:
assumes IsAnormalSubgroup(G,P,H) g∈G
shows –g·(h·g^{-1}). h∈H″⊆H
⟨*proof*⟩

If a subgroup contains all its conjugate subgroups, then it is normal.

theorem (in group0) cont˙conj˙is˙normal:
assumes IsAsubgroup(H,P) ∀g∈G. –g·(h·g^{-1}). h∈H″⊆H
shows IsAnormalSubgroup(G,P,H)
⟨*proof*⟩

If a group has only one subgroup of a given order, then this subgroup is normal.

corollary(in group0) only˙one˙equipoll˙sub:
assumes IsAsubgroup(H,P) ∀M. IsAsubgroup(M,P)∧ H≈M ⟶ M=H
shows IsAnormalSubgroup(G,P,H)
⟨*proof*⟩

The trivial subgroup is then a normal subgroup.

corollary(in group0) trivial˙normal˙subgroup:
shows IsAnormalSubgroup(G,P,–**1**″)
⟨*proof*⟩

lemma(in group0) whole˙normal˙subgroup:
shows IsAnormalSubgroup(G,P,G)
⟨*proof*⟩

Since the whole group and the trivial subgroup are normal, it is natural to define simplicity of groups in the following way:

definition
IsSimple ([˙,˙]–is a simple group″ 89)
where [G,f]–is a simple group″ ≡ IsAgroup(G,f)∧(∀M. IsAnormalSubgroup(G,f,M) ⟶ M=G∨M=–TheNeutralElement(G,f)″)

From the definition follows that if a group has no subgroups, then it is simple.

corollary (in group0) noSubgroup˙imp˙simple:
  assumes ∀ H. IsAsubgroup(H,P) ⟶ H=G∨H=–**1**″
  shows [G,P]–is a simple group″
⟨*proof*⟩

Since every subgroup is normal in abelian groups, it follows that commutative simple groups do not have subgroups.

corollary (in group0) abelian˙simple˙noSubgroups:
  assumes [G,P]–is a simple group″ P–is commutative on″G
  shows ∀ H. IsAsubgroup(H,P) ⟶ H=G∨H=–**1**″
⟨*proof*⟩

39.2 Finite groups

The subgroup of a finite group is finite.

lemma(in group0) finite˙subgroup:
  assumes Finite(G) IsAsubgroup(H,P)
  shows Finite(H)
  ⟨*proof*⟩

The space of cosets is also finite. In particular, quotient groups.

lemma(in group0) finite˙cosets:
  assumes Finite(G) IsAsubgroup(H,P) r=QuotientGroupRel(G,P,H)
  shows Finite(G//r)
⟨*proof*⟩

All the cosets are equipollent.

lemma(in group0) cosets˙equipoll:
  assumes IsAsubgroup(H,P) r=QuotientGroupRel(G,P,H) g1∈Gg2∈G
  shows r–g1″≈r–g2″
⟨*proof*⟩

The order of a subgroup multiplied by the order of the space of cosets is the order of the group. We only prove the theorem for finite groups.

theorem(in group0) Lagrange:
  assumes Finite(G) IsAsubgroup(H,P) r=QuotientGroupRel(G,P,H)
  shows —G—=—H— #* —G//r—
⟨*proof*⟩

39.3 Subgroups generated by sets

Given a subset of a group, we can ask ourselves which is the smallest group that contains that set; if it even exists.

lemma(in group0) inter˙subgroups:
  assumes ∀ H∈$\mathfrak{H}$. IsAsubgroup(H,P) $\mathfrak{H}$≠0
  shows IsAsubgroup(⋂ $\mathfrak{H}$,P)
⟨*proof*⟩

As the previous lemma states, the subgroup that contains a subset can be defined as an intersection of subgroups.

definition(in group0)
  SubgroupGenerated ($\langle \dot{} \rangle_G$ 80)
  where $\langle X \rangle_G \equiv \bigcap$–H∈Pow(G). X⊆H ∧ IsAsubgroup(H,P)˝

theorem(in group0) subgroupGen˙is˙subgroup:
  assumes X⊆G
  shows IsAsubgroup($\langle X \rangle_G$,P)
⟨*proof*⟩

39.4 Homomorphisms

A homomorphism is a function between groups that preserves group operations.

definition
  Homomor (˙–is a homomorphism˝–˙,˙˝→–˙,˙˝ 85)
  where IsAgroup(G,P) ⟹ IsAgroup(H,F) ⟹ Homomor(f,G,P,H,F) ≡ ∀g1∈G. ∀g2∈G. f(P⟨g1,g2⟩)=F⟨fg1,fg2⟩

Now a lemma about the definition:

lemma homomor˙eq:
  assumes IsAgroup(G,P) IsAgroup(H,F) Homomor(f,G,P,H,F) g1∈G g2∈G
  shows f(P⟨g1,g2⟩)=F⟨fg1,fg2⟩
  ⟨*proof*⟩

An endomorphism is a homomorphism from a group to the same group. In case the group is abelian, it has a nice structure.

definition
  End
  where End(G,P) ≡ –f:G→G. Homomor(f,G,P,G,P)˝

The set of endomorphisms forms a submonoid of the monoid of function from a set to that set under composition.

lemma(in group0) end˙composition:
  assumes f1∈End(G,P)f2∈End(G,P)
  shows Composition(G)⟨f1,f2⟩∈End(G,P)
⟨*proof*⟩

theorem(in group0) end˙comp˙monoid:
  shows IsAmonoid(End(G,P),restrict(Composition(G),End(G,P)×End(G,P)))
 and TheNeutralElement(End(G,P),restrict(Composition(G),End(G,P)×End(G,P)))=id(G)
⟨*proof*⟩

The set of endomorphisms is closed under pointwise addition. This is so because the group is abelian.

theorem(in group0) end˙pointwise˙addition:

assumes f∈End(G,P)g∈End(G,P)P–is commutative on″GF = P –lifted to function space over″ G
shows F⟨f,g⟩∈End(G,P)
⟨*proof*⟩

The inverse of an abelian group is an endomorphism.

lemma(in group0) end˙inverse˙group:
assumes P–is commutative on″G
shows GroupInv(G,P)∈End(G,P)
⟨*proof*⟩

The set of homomorphisms of an abelian group is an abelian subgroup of the group of functions from a set to a group, under pointwise multiplication.

theorem(in group0) end˙addition˙group:
assumes P–is commutative on″G F = P –lifted to function space over″ G
shows IsAgroup(End(G,P),restrict(F,End(G,P)×End(G,P))) restrict(F,End(G,P)×End(G,P))–is commutative on″End(G,P)
⟨*proof*⟩

lemma(in group0) distributive˙comp˙pointwise:
assumes P–is commutative on″G F = P –lifted to function space over″ G
shows IsDistributive(End(G,P),restrict(F,End(G,P)×End(G,P)),restrict(Composition(G),End(G,P)×End(G,
⟨*proof*⟩

The endomorphisms of an abelian group is in fact a ring with the previous operations.

theorem(in group0) end˙is˙ring:
assumes P–is commutative on″G F = P –lifted to function space over″ G
shows IsAring(End(G,P),restrict(F,End(G,P)×End(G,P)),restrict(Composition(G),End(G,P)×End(G,P)))
⟨*proof*⟩

39.5 First isomorphism theorem

Now we will prove that any homomorphism $f : G \to H$ defines a bijective homomorphism between G/H and $f(G)$.

A group homomorphism sends the neutral element to the neutral element and commutes with the inverse.

lemma image˙neutral:
assumes IsAgroup(G,P) IsAgroup(H,F) Homomor(f,G,P,H,F) f:G→H
shows fTheNeutralElement(G,P)=TheNeutralElement(H,F)
⟨*proof*⟩

lemma image˙inv:
assumes IsAgroup(G,P) IsAgroup(H,F) Homomor(f,G,P,H,F) f:G→H g∈G
shows f(GroupInv(G,P)g)=GroupInv(H,F) (fg)
⟨*proof*⟩

The kernel of an homomorphism is a normal subgroup.

theorem kerner˙normal˙sub:
  assumes IsAgroup(G,P) IsAgroup(H,F) Homomor(f,G,P,H,F) f:G→H
  shows IsAnormalSubgroup(G,P,f-–TheNeutralElement(H,F)˝)
⟨*proof*⟩

The image of a homomorphism is a subgroup.

theorem image˙sub:
  assumes IsAgroup(G,P) IsAgroup(H,F) Homomor(f,G,P,H,F) f:G→H
  shows IsAsubgroup(fG,F)
⟨*proof*⟩

Now we are able to prove the first isomorphism theorem. This theorem states that any group homomorphism $f : G \to H$ gives an isomorphism between a quotient group of G and a subgroup of H.

theorem isomorphism˙first˙theorem:
  assumes IsAgroup(G,P) IsAgroup(H,F) Homomor(f,G,P,H,F) f:G→H
  defines r ≡ QuotientGroupRel(G,P,f-–TheNeutralElement(H,F)˝) and
  PP ≡ QuotientGroupOp(G,P,f-–TheNeutralElement(H,F)˝)
  shows ∃ff. Homomor(ff,G//r,PP,fG,restrict(F,(fG)×(fG))) ∧ ff∈bij(G//r,fG)
⟨*proof*⟩

As a last result, the inverse of a bijective homomorphism is an homomorphism. Meaning that in the previous result, the homomorphism we found is an isomorphism.

theorem bij˙homomor:
  assumes f∈bij(G,H)IsAgroup(G,P)IsAgroup(H,F)Homomor(f,G,P,H,F)
  shows Homomor(converse(f),H,F,G,P)
⟨*proof*⟩

end

40 Fields - introduction

theory Field˙ZF imports Ring˙ZF

begin

This theory covers basic facts about fields.

40.1 Definition and basic properties

In this section we define what is a field and list the basic properties of fields.

Field is a notrivial commutative ring such that all non-zero elements have an inverse. We define the notion of being a field as a statement about three sets. The first set, denoted K is the carrier of the field. The second set, denoted A represents the additive operation on K (recall that in ZF set theory functions are sets). The third set M represents the multiplicative operation on K.

definition
  IsAfield(K,A,M) ≡
  (IsAring(K,A,M) ∧ (M –is commutative on″ K) ∧
  TheNeutralElement(K,A) ≠ TheNeutralElement(K,M) ∧
  (∀ a∈K. a≠TheNeutralElement(K,A)⟶
  (∃ b∈K. M⟨a,b⟩ = TheNeutralElement(K,M))))

The field0 context extends the ring0 context adding field-related assumptions and notation related to the multiplicative inverse.

locale field0 = ring0 K A M for K A M +
  assumes mult˙commute: M –is commutative on″ K

  assumes not˙triv: **0** ≠ **1**

  assumes inv˙exists: ∀ a∈K. a≠**0** ⟶ (∃ b∈K. a·b = **1**)

  fixes non˙zero (K_0)
  defines non˙zero˙def[simp]: K_0 ≡ K-–**0**″

  fixes inv (˙$^{-1}$ [96] 97)
  defines inv˙def[simp]: a^{-1} ≡ GroupInv(K_0,restrict(M,K_0×K_0))(a)

The next lemma assures us that we are talking fields in the field0 context.

lemma (in field0) Field˙ZF˙1˙L1: shows IsAfield(K,A,M)
  ⟨*proof*⟩

We can use theorems proven in the field0 context whenever we talk about a field.

lemma field˙field0: assumes IsAfield(K,A,M)
  shows field0(K,A,M)
  ⟨*proof*⟩

Let's have an explicit statement that the multiplication in fields is commutative.

lemma (in field0) field˙mult˙comm: assumes a∈K b∈K
  shows a·b = b·a
  ⟨*proof*⟩

Fields do not have zero divisors.

lemma (in field0) field˙has˙no˙zero˙divs: shows HasNoZeroDivs(K,A,M)
⟨*proof*⟩

K_0 (the set of nonzero field elements is closed with respect to multiplication.

lemma (in field0) Field˙ZF˙1˙L2:
  shows K_0 –is closed under″ M

$\langle proof \rangle$

Any nonzero element has a right inverse that is nonzero.

lemma (in field0) Field_ZF_1_L3: assumes A1: $a \in K_0$
shows $\exists b \in K_0.\ a \cdot b = \mathbf{1}$
$\langle proof \rangle$

If we remove zero, the field with multiplication becomes a group and we can use all theorems proven in group0 context.

theorem (in field0) Field_ZF_1_L4: shows
IsAgroup(K_0,restrict(M,$K_0 \times K_0$))
group0(K_0,restrict(M,$K_0 \times K_0$))
$\mathbf{1}$ = TheNeutralElement(K_0,restrict(M,$K_0 \times K_0$))
$\langle proof \rangle$

The inverse of a nonzero field element is nonzero.

lemma (in field0) Field_ZF_1_L5: assumes A1: $a \in K$ $a \neq \mathbf{0}$
shows $a^{-1} \in K_0$ $(a^{-1})^2 \in K_0$ $a^{-1} \in K$ $a^{-1} \neq \mathbf{0}$
$\langle proof \rangle$

The inverse is really the inverse.

lemma (in field0) Field_ZF_1_L6: assumes A1: $a \in K$ $a \neq \mathbf{0}$
shows $a \cdot a^{-1} = \mathbf{1}$ $a^{-1} \cdot a = \mathbf{1}$
$\langle proof \rangle$

A lemma with two field elements and cancelling.

lemma (in field0) Field_ZF_1_L7: assumes $a \in K$ $b \in K$ $b \neq \mathbf{0}$
shows
$a \cdot b \cdot b^{-1} = a$
$a \cdot b^{-1} \cdot b = a$
$\langle proof \rangle$

40.2 Equations and identities

This section deals with more specialized identities that are true in fields.

$a/(a^2) = 1/a$.

lemma (in field0) Field_ZF_2_L1: assumes A1: $a \in K$ $a \neq \mathbf{0}$
shows $a \cdot (a^{-1})^2 = a^{-1}$
$\langle proof \rangle$

If we multiply two different numbers by a nonzero number, the results will be different.

lemma (in field0) Field_ZF_2_L2:
assumes $a \in K$ $b \in K$ $c \in K$ $a \neq b$ $c \neq \mathbf{0}$
shows $a \cdot c^{-1} \neq b \cdot c^{-1}$
$\langle proof \rangle$

We can put a nonzero factor on the other side of non-identity (is this the best way to call it?) changing it to the inverse.

lemma (in field0) Field_ZF_2_L3:
  assumes A1: a∈K b∈K b≠**0** c∈K   and A2: a·b ≠ c
  shows a ≠ c·b^{-1}
⟨*proof*⟩

If if the inverse of b is different than a, then the inverse of a is different than b.

lemma (in field0) Field_ZF_2_L4:
  assumes a∈K a≠**0** and b^{-1} ≠ a
  shows a^{-1} ≠ b
  ⟨*proof*⟩

An identity with two field elements, one and an inverse.

lemma (in field0) Field_ZF_2_L5:
  assumes a∈K b∈K b≠**0**
  shows (**1** + a·b)·b^{-1} = a + b^{-1}
  ⟨*proof*⟩

An identity with three field elements, inverse and cancelling.

lemma (in field0) Field_ZF_2_L6: assumes A1: a∈K b∈K b≠**0** c∈K
  shows a·b·(c·b^{-1}) = a·c
⟨*proof*⟩

40.3 1/0=0

In ZF if $f : X \to Y$ and $x \notin X$ we have $f(x) = \emptyset$. Since $\emptyset$ (the empty set) in ZF is the same as zero of natural numbers we can claim that $1/0 = 0$ in certain sense. In this section we prove a theorem that makes makes it explicit.

The next locale extends the field0 locale to introduce notation for division operation.

locale fieldd = field0 +
  fixes division
  defines division_def[simp]: division ≡ {⟨p,fst(p)·snd(p)$^{-1}$⟩. p∈K×K$_0$}

  fixes fdiv (infixl / 95)
  defines fdiv_def[simp]: x/y ≡ division⟨x,y⟩

Division is a function on $K \times K_0$ with values in K.

lemma (in fieldd) div_fun: shows division: K×K$_0$ → K
⟨*proof*⟩

So, really $1/0 = 0$. The essential lemma is apply_0 from standard Isabelle's func.thy.

theorem (in fieldd) one˙over˙zero: shows $\mathbf{1}/\mathbf{0} = 0$
⟨*proof*⟩

end

41 Ordered fields

theory OrderedField˙ZF imports OrderedRing˙ZF Field˙ZF

begin

This theory covers basic facts about ordered fiels.

41.1 Definition and basic properties

Here we define ordered fields and proove their basic properties.

Ordered field is a notrivial ordered ring such that all non-zero elements have an inverse. We define the notion of being a ordered field as a statement about four sets. The first set, denoted K is the carrier of the field. The second set, denoted A represents the additive operation on K (recall that in ZF set theory functions are sets). The third set M represents the multiplicative operation on K. The fourth set r is the order relation on K.

definition
  IsAnOrdField(K,A,M,r) ≡ (IsAnOrdRing(K,A,M,r) ∧
  (M –is commutative on″ K) ∧
  TheNeutralElement(K,A) ≠ TheNeutralElement(K,M) ∧
  (∀ a∈K. a≠TheNeutralElement(K,A)⟶
  (∃ b∈K. M⟨a,b⟩ = TheNeutralElement(K,M))))

The next context (locale) defines notation used for ordered fields. We do that by extending the notation defined in the ring1 context that is used for oredered rings and adding some assumptions to make sure we are talking about ordered fields in this context. We should rename the carrier from R used in the ring1 context to K, more appriopriate for fields. Theoretically the Isar locale facility supports such renaming, but we experienced diffculties using some lemmas from ring1 locale after renaming.

locale field1 = ring1 +

  assumes mult˙commute: M –is commutative on″ R

  assumes not˙triv: $\mathbf{0} \neq \mathbf{1}$

  assumes inv˙exists: ∀ a∈R. a≠$\mathbf{0}$ ⟶ (∃ b∈R. a·b = $\mathbf{1}$)

  fixes non˙zero (R_0)

defines non˙zero˙def[simp]: $R_0 \equiv R$-–$\mathbf{0}$″

fixes inv (˙$^{-1}$ [96] 97)
defines inv˙def[simp]: $a^{-1} \equiv$ GroupInv(R_0,restrict(M,$R_0 \times R_0$))(a)

The next lemma assures us that we are talking fields in the field1 context.

lemma (in field1) OrdField˙ZF˙1˙L1: shows IsAnOrdField(R,A,M,r)
⟨*proof*⟩

Ordered field is a field, of course.

lemma OrdField˙ZF˙1˙L1A: assumes IsAnOrdField(K,A,M,r)
shows IsAfield(K,A,M)
⟨*proof*⟩

Theorems proven in field0 (about fields) context are valid in the field1 context (about ordered fields).

lemma (in field1) OrdField˙ZF˙1˙L1B: shows field0(R,A,M)
⟨*proof*⟩

We can use theorems proven in the field1 context whenever we talk about an ordered field.

lemma OrdField˙ZF˙1˙L2: assumes IsAnOrdField(K,A,M,r)
shows field1(K,A,M,r)
⟨*proof*⟩

In ordered rings the existence of a right inverse for all positive elements implies the existence of an inverse for all non zero elements.

lemma (in ring1) OrdField˙ZF˙1˙L3:
assumes A1: $\forall a \in R_+$. $\exists b \in R$. $a \cdot b = \mathbf{1}$ and A2: $c \in R$ $c \neq \mathbf{0}$
shows $\exists b \in R$. $c \cdot b = \mathbf{1}$
⟨*proof*⟩

Ordered fields are easier to deal with, because it is sufficient to show the existence of an inverse for the set of positive elements.

lemma (in ring1) OrdField˙ZF˙1˙L4:
assumes $\mathbf{0} \neq \mathbf{1}$ and M –is commutative on″ R
and $\forall a \in R_+$. $\exists b \in R$. $a \cdot b = \mathbf{1}$
shows IsAnOrdField(R,A,M,r)
⟨*proof*⟩

The set of positive field elements is closed under multiplication.

lemma (in field1) OrdField˙ZF˙1˙L5: shows R_+ –is closed under″ M
⟨*proof*⟩

The set of positive field elements is closed under multiplication: the explicit version.

lemma (in field1) pos˙mul˙closed:

assumes A1: **0** ¡ a **0** ¡ b
shows **0** ¡ a·b
⟨*proof*⟩

In fields square of a nonzero element is positive.

lemma (in field1) OrdField˙ZF˙1˙L6: assumes a∈R a≠**0**
shows $a^2 \in R_+$
⟨*proof*⟩

The next lemma restates the fact Field˙ZF that out notation for the field inverse means what it is supposed to mean.

lemma (in field1) OrdField˙ZF˙1˙L7: assumes a∈R a≠**0**
shows $a\cdot(a^{-1}) = \mathbf{1}$ $(a^{-1})\cdot a = \mathbf{1}$
⟨*proof*⟩

A simple lemma about multiplication and cancelling of a positive field element.

lemma (in field1) OrdField˙ZF˙1˙L7A:
assumes A1: a∈R $b \in R_+$
shows
$a\cdot b\cdot b^{-1} = a$
$a\cdot b^{-1}\cdot b = a$
⟨*proof*⟩

Some properties of the inverse of a positive element.

lemma (in field1) OrdField˙ZF˙1˙L8: assumes A1: $a \in R_+$
shows $a^{-1} \in R_+$ $a\cdot(a^{-1}) = \mathbf{1}$ $(a^{-1})\cdot a = \mathbf{1}$
⟨*proof*⟩

If $a < b$, then $(b - a)^{-1}$ is positive.

lemma (in field1) OrdField˙ZF˙1˙L9: assumes a¡b
shows $(b\text{-}a)^{-1} \in R_+$
⟨*proof*⟩

In ordered fields if at least one of a, b is not zero, then $a^2 + b^2 > 0$, in particular $a^2 + b^2 \neq 0$ and exists the (multiplicative) inverse of $a^2 + b^2$.

lemma (in field1) OrdField˙ZF˙1˙L10:
assumes A1: a∈R b∈R and A2: a ≠ **0** ∨ b ≠ **0**
shows **0** ¡ $a^2 + b^2$ and ∃c∈R. $(a^2 + b^2)\cdot c = \mathbf{1}$
⟨*proof*⟩

41.2 Inequalities

In this section we develop tools to deal inequalities in fields.

We can multiply strict inequality by a positive element.

lemma (in field1) OrdField˙ZF˙2˙L1:

assumes a¡b and c∈R_+
shows a·c ¡ b·c
⟨*proof*⟩

A special case of OrdField˙ZF˙2˙L1 when we multiply an inverse by an element.

lemma (in field1) OrdField˙ZF˙2˙L2:
assumes A1: a∈R_+ and A2: a^{-1} ¡ b
shows **1** ¡ b·a
⟨*proof*⟩

We can multiply an inequality by the inverse of a positive element.

lemma (in field1) OrdField˙ZF˙2˙L3:
assumes a≤b and c∈R_+ shows a·(c^{-1}) ≤ b·(c^{-1})
⟨*proof*⟩

We can multiply a strict inequality by a positive element or its inverse.

lemma (in field1) OrdField˙ZF˙2˙L4:
assumes a¡b and c∈R_+
shows
a·c ¡ b·c
c·a ¡ c·b
a·c^{-1} ¡ b·c^{-1}
⟨*proof*⟩

We can put a positive factor on the other side of an inequality, changing it to its inverse.

lemma (in field1) OrdField˙ZF˙2˙L5:
assumes A1: a∈R b∈R_+ and A2: a·b ≤ c
shows a ≤ c·b^{-1}
⟨*proof*⟩

We can put a positive factor on the other side of an inequality, changing it to its inverse, version with a product initially on the right hand side.

lemma (in field1) OrdField˙ZF˙2˙L5A:
assumes A1: b∈R c∈R_+ and A2: a ≤ b·c
shows a·c^{-1} ≤ b
⟨*proof*⟩

We can put a positive factor on the other side of a strict inequality, changing it to its inverse, version with a product initially on the left hand side.

lemma (in field1) OrdField˙ZF˙2˙L6:
assumes A1: a∈R b∈R_+ and A2: a·b ¡ c
shows a ¡ c·b^{-1}
⟨*proof*⟩

We can put a positive factor on the other side of a strict inequality, changing it to its inverse, version with a product initially on the right hand side.

lemma (in field1) OrdField˙ZF˙2˙L6A:
  assumes A1: b∈R c∈R_+ and A2: a ¡ b·c
  shows a·c^{-1} ¡ b
⟨*proof*⟩

Sometimes we can reverse an inequality by taking inverse on both sides.

lemma (in field1) OrdField˙ZF˙2˙L7:
  assumes A1: a∈R_+ and A2: a^{-1} ≤ b
  shows b^{-1} ≤ a
⟨*proof*⟩

Sometimes we can reverse a strict inequality by taking inverse on both sides.

lemma (in field1) OrdField˙ZF˙2˙L8:
  assumes A1: a∈R_+ and A2: a^{-1} ¡ b
  shows b^{-1} ¡ a
⟨*proof*⟩

A technical lemma about solving a strict inequality with three field elements and inverse of a difference.

lemma (in field1) OrdField˙ZF˙2˙L9:
  assumes A1: a¡b and A2: (b-a)$^{-1}$ ¡ c
  shows **1** + a·c ¡ b·c
⟨*proof*⟩

41.3 Definition of real numbers

The only purpose of this section is to define what does it mean to be a model of real numbers.

We define model of real numbers as any quadruple of sets (K, A, M, r) such that (K, A, M, r) is an ordered field and the order relation r is complete, that is every set that is nonempty and bounded above in this relation has a supremum.

definition
  IsAmodelOfReals(K,A,M,r) ≡ IsAnOrdField(K,A,M,r) ∧ (r –is complete˝)

end

42 Integers - introduction

theory Int˙ZF˙IML imports OrderedGroup˙ZF˙1 Finite˙ZF˙1 ZF.Int Nat˙ZF˙IML

begin

This theory file is an interface between the old-style Isabelle (ZF logic) material on integers and the IsarMathLib project. Here we redefine the meta-level operations on integers (addition and multiplication) to convert them to ZF-functions and show that integers form a commutative group with respect to addition and commutative monoid with respect to multiplication. Similarly, we redefine the order on integers as a relation, that is a subset of $Z \times Z$. We show that a subset of intergers is bounded iff it is finite. As we are forced to use standard Isabelle notation with all these dollar signs, sharps etc. to denote "type coercions" (?) the notation is often ugly and difficult to read.

42.1 Addition and multiplication as ZF-functions.

In this section we provide definitions of addition and multiplication as subsets of $(Z \times Z) \times Z$. We use the (higher order) relation defined in the standard Int theory to define a subset of $Z \times Z$ that constitutes the ZF order relation corresponding to it. We define the set of positive integers using the notion of positive set from the OrderedGroup˙ZF theory.

Definition of addition of integers as a binary operation on int. Recall that in standard Isabelle/ZF int is the set of integers and the sum of integers is denoted by prependig + with a dollar sign.

definition
  IntegerAddition ≡ – ⟨ x,c⟩ ∈ (int×int)×int. fst(x) $+ snd(x) = c″

Definition of multiplication of integers as a binary operation on int. In standard Isabelle/ZF product of integers is denoted by prepending the dollar sign to *.

definition
  IntegerMultiplication ≡
    – ⟨ x,c⟩ ∈ (int×int)×int. fst(x) $* snd(x) = c″

Definition of natural order on integers as a relation on int. In the standard Isabelle/ZF the inequality relation on integers is denoted ≤ prepended with the dollar sign.

definition
  IntegerOrder ≡ –p ∈ int×int. fst(p) $≤ snd(p)″

This defines the set of positive integers.

definition
  PositiveIntegers ≡ PositiveSet(int,IntegerAddition,IntegerOrder)

IntegerAddition and IntegerMultiplication are functions on int × int.

lemma Int˙ZF˙1˙L1: shows
  IntegerAddition : int×int → int
  IntegerMultiplication : int×int → int
⟨*proof*⟩

The next context (locale) defines notation used for integers. We define **0** to denote the neutral element of addition, **1** as the unit of the multiplicative monoid. We introduce notation m≤n for integers and write m..n to denote the integer interval with endpoints in m and n. abs(m) means the absolute value of m. This is a function defined in OrderedGroup that assigns x to itself if x is positive and assigns the opposite of x if $x \leq 0$. Unforunately we cannot use the $|\cdot|$ notation as in the OrderedGroup theory as this notation has been hogged by the standard Isabelle's Int theory. The notation -A where A is a subset of integers means the set $\{-m : m \in A\}$. The symbol maxf(f,M) denotes tha maximum of function f over the set A. We also introduce a similar notation for the minimum.

locale int0 =

fixes ints ($\mathbb{Z}$)
defines ints˙def [simp]: $\mathbb{Z}$ ≡ int

fixes ia (infixl + 69)
defines ia˙def [simp]: a+b ≡ IntegerAddition⟨ a,b⟩

fixes iminus (- ˙ 72)
defines rminus˙def [simp]: -a ≡ GroupInv($\mathbb{Z}$,IntegerAddition)(a)

fixes isub (infixl - 69)
defines isub˙def [simp]: a-b ≡ a+ (- b)

fixes imult (infixl · 70)
defines imult˙def [simp]: a·b ≡ IntegerMultiplication⟨ a,b⟩

fixes setneg (- ˙ 72)
defines setneg˙def [simp]: -A ≡ GroupInv($\mathbb{Z}$,IntegerAddition)(A)

fixes izero (**0**)
defines izero˙def [simp]: **0** ≡ TheNeutralElement($\mathbb{Z}$,IntegerAddition)

fixes ione (**1**)
defines ione˙def [simp]: **1** ≡ TheNeutralElement($\mathbb{Z}$,IntegerMultiplication)

fixes itwo (**2**)
defines itwo˙def [simp]: **2** ≡ **1**+**1**

fixes ithree (**3**)
defines ithree˙def [simp]: **3** ≡ **2**+**1**

fixes nonnegative ($\mathbb{Z}^+$)
defines nonnegative˙def [simp]:
$\mathbb{Z}^+$ ≡ Nonnegative($\mathbb{Z}$,IntegerAddition,IntegerOrder)

fixes positive ($\mathbb{Z}_+$)

defines positive˙def [simp]:
$\mathbb{Z}_+ \equiv$ PositiveSet($\mathbb{Z}$,IntegerAddition,IntegerOrder)

fixes abs
defines abs˙def [simp]:
abs(m) $\equiv$ AbsoluteValue($\mathbb{Z}$,IntegerAddition,IntegerOrder)(m)

fixes lesseq (infix $\leq$ 60)
defines lesseq˙def [simp]: m $\leq$ n $\equiv$ $\langle$m,n$\rangle$ $\in$ IntegerOrder

fixes interval (infix .. 70)
defines interval˙def [simp]: m..n $\equiv$ Interval(IntegerOrder,m,n)

fixes maxf
defines maxf˙def [simp]: maxf(f,A) $\equiv$ Maximum(IntegerOrder,f(A))

fixes minf
defines minf˙def [simp]: minf(f,A) $\equiv$ Minimum(IntegerOrder,f(A))

IntegerAddition adds integers and IntegerMultiplication multiplies integers. This states that the ZF functions IntegerAddition and IntegerMultiplication give the same results as the higher-order equivalents defined in the standard Int theory.

lemma (in int0) Int˙ZF˙1˙L2: assumes A1: a $\in \mathbb{Z}$ b $\in \mathbb{Z}$
shows
a+b = a $+ b
a·b = a $* b
⟨*proof*⟩

Integer addition and multiplication are associative.

lemma (in int0) Int˙ZF˙1˙L3:
assumes x$\in\mathbb{Z}$ y$\in\mathbb{Z}$ z$\in\mathbb{Z}$
shows x+y+z = x+(y+z) x·y·z = x·(y·z)
⟨*proof*⟩

Integer addition and multiplication are commutative.

lemma (in int0) Int˙ZF˙1˙L4:
assumes x$\in\mathbb{Z}$ y$\in\mathbb{Z}$
shows x+y = y+x x·y = y·x
⟨*proof*⟩

Zero is neutral for addition and one for multiplication.

lemma (in int0) Int˙ZF˙1˙L5: assumes A1:x$\in\mathbb{Z}$
shows ($# 0) + x = x $\wedge$ x + ($# 0) = x
($# 1)·x = x $\wedge$ x·($# 1) = x
⟨*proof*⟩

Zero is neutral for addition and one for multiplication.

lemma (in int0) Int˙ZF˙1˙L6: shows ($# 0)∈$\mathbb{Z}$ ∧
  (∀x∈$\mathbb{Z}$. ($# 0)+x = x ∧ x+($# 0) = x)
  ($# 1)∈$\mathbb{Z}$ ∧
  (∀x∈$\mathbb{Z}$. ($# 1)·x = x ∧ x·($# 1) = x)
  ⟨*proof*⟩

Integers with addition and integers with multiplication form monoids.

theorem (in int0) Int˙ZF˙1˙T1: shows
  IsAmonoid($\mathbb{Z}$,IntegerAddition)
  IsAmonoid($\mathbb{Z}$,IntegerMultiplication)
⟨*proof*⟩

Zero is the neutral element of the integers with addition and one is the neutral element of the integers with multiplication.

lemma (in int0) Int˙ZF˙1˙L8: shows ($# 0) = **0** ($# 1) = **1**
⟨*proof*⟩

0 and 1, as defined in int0 context, are integers.

lemma (in int0) Int˙ZF˙1˙L8A: shows **0** ∈ $\mathbb{Z}$ **1** ∈ $\mathbb{Z}$
⟨*proof*⟩

Zero is not one.

lemma (in int0) int˙zero˙not˙one: shows **0** ≠ **1**
⟨*proof*⟩

The set of integers is not empty, of course.

lemma (in int0) int˙not˙empty: shows $\mathbb{Z}$ ≠ 0
  ⟨*proof*⟩

The set of integers has more than just zero in it.

lemma (in int0) int˙not˙trivial: shows $\mathbb{Z}$ ≠ –**0**″
  ⟨*proof*⟩

Each integer has an inverse (in the addition sense).

lemma (in int0) Int˙ZF˙1˙L9: assumes A1: g ∈ $\mathbb{Z}$
  shows ∃ b∈$\mathbb{Z}$. g+b = **0**
⟨*proof*⟩

Integers with addition form an abelian group. This also shows that we can apply all theorems proven in the proof contexts (locales) that require the assumpion that some pair of sets form a group like locale group0.

theorem Int˙ZF˙1˙T2: shows
  IsAgroup(int,IntegerAddition)
  IntegerAddition –is commutative on″ int
  group0(int,IntegerAddition)
  ⟨*proof*⟩

What is the additive group inverse in the group of integers?

lemma (in int0) Int_ZF_1_L9A: assumes A1: m∈ℤ
  shows $-m = -m
⟨proof⟩

Subtracting integers corresponds to adding the negative.

lemma (in int0) Int_ZF_1_L10: assumes A1: m∈ℤ n∈ℤ
  shows m-n = m $+ $-n
  ⟨proof⟩

Negative of zero is zero.

lemma (in int0) Int_ZF_1_L11: shows (-**0**) = **0**
  ⟨proof⟩

A trivial calculation lemma that allows to subtract and add one.

lemma Int_ZF_1_L12:
  assumes m∈int shows m $- $#1 $+ $#1 = m
  ⟨proof⟩

A trivial calculation lemma that allows to subtract and add one, version with ZF-operation.

lemma (in int0) Int_ZF_1_L13: assumes m∈ℤ
  shows (m $- $#1) + **1** = m
  ⟨proof⟩

Adding or subtracing one changes integers.

lemma (in int0) Int_ZF_1_L14: assumes A1: m∈ℤ
  shows
  m+**1** ≠ m
  m-**1** ≠ m
⟨proof⟩

If the difference is zero, the integers are equal.

lemma (in int0) Int_ZF_1_L15:
  assumes A1: m∈ℤ n∈ℤ and A2: m-n = **0**
  shows m=n
⟨proof⟩

42.2 Integers as an ordered group

In this section we define order on integers as a relation, that is a subset of $Z \times Z$ and show that integers form an ordered group.

The next lemma interprets the order definition one way.

lemma (in int0) Int_ZF_2_L1:
  assumes A1: m∈ℤ n∈ℤ and A2: m $≤ n

shows m ≤ n
⟨*proof*⟩

The next lemma interprets the definition the other way.

lemma (in int0) Int˙ZF˙2˙L1A: assumes A1: m ≤ n
shows m $≤ n m∈ℤ n∈ℤ
⟨*proof*⟩

Integer order is a relation on integers.

lemma Int˙ZF˙2˙L1B: shows IntegerOrder ⊆ int×int
⟨*proof*⟩

The way we define the notion of being bounded below, its sufficient for the relation to be on integers for all bounded below sets to be subsets of integers.

lemma (in int0) Int˙ZF˙2˙L1C:
assumes A1: IsBoundedBelow(A,IntegerOrder)
shows A⊆ℤ
⟨*proof*⟩

The order on integers is reflexive.

lemma (in int0) int˙ord˙is˙refl: shows refl(ℤ,IntegerOrder)
⟨*proof*⟩

The essential condition to show antisymmetry of the order on integers.

lemma (in int0) Int˙ZF˙2˙L3:
assumes A1: m ≤ n n ≤ m
shows m=n
⟨*proof*⟩

The order on integers is antisymmetric.

lemma (in int0) Int˙ZF˙2˙L4: shows antisym(IntegerOrder)
⟨*proof*⟩

The essential condition to show that the order on integers is transitive.

lemma Int˙ZF˙2˙L5:
assumes A1: ⟨m,n⟩ ∈ IntegerOrder ⟨n,k⟩ ∈ IntegerOrder
shows ⟨m,k⟩ ∈ IntegerOrder
⟨*proof*⟩

The order on integers is transitive. This version is stated in the int0 context using notation for integers.

lemma (in int0) Int˙order˙transitive:
assumes A1: m≤n n≤k
shows m≤k
⟨*proof*⟩

The order on integers is transitive.

lemma Int˙ZF˙2˙L6: shows trans(IntegerOrder)
⟨*proof*⟩

The order on integers is a partial order.

lemma Int˙ZF˙2˙L7: shows IsPartOrder(int,IntegerOrder)
⟨*proof*⟩

The essential condition to show that the order on integers is preserved by translations.

lemma (in int0) int˙ord˙transl˙inv:
assumes A1: k ∈ ℤ and A2: m ≤ n
shows m+k ≤ n+k k+m≤ k+n
⟨*proof*⟩

Integers form a linearly ordered group. We can apply all theorems proven in group3 context to integers.

theorem (in int0) Int˙ZF˙2˙T1: shows
IsAnOrdGroup(ℤ,IntegerAddition,IntegerOrder)
IntegerOrder –is total on″ ℤ
group3(ℤ,IntegerAddition,IntegerOrder)
IsLinOrder(ℤ,IntegerOrder)
⟨*proof*⟩

If a pair (i, m) belongs to the order relation on integers and $i \neq m$, then $i < m$ in the sense of defined in the standard Isabelle's Int.thy.

lemma (in int0) Int˙ZF˙2˙L9: assumes A1: i ≤ m and A2: i≠m
shows i $¡ m
⟨*proof*⟩

This shows how Isabelle's $¡ operator translates to IsarMathLib notation.

lemma (in int0) Int˙ZF˙2˙L9AA: assumes A1: m∈ℤ n∈ℤ
and A2: m $¡ n
shows m≤n m ≠ n
⟨*proof*⟩

A small technical lemma about putting one on the other side of an inequality.

lemma (in int0) Int˙ZF˙2˙L9A:
assumes A1: k∈ℤ and A2: m ≤ k $- ($# 1)
shows m+**1** ≤ k
⟨*proof*⟩

We can put any integer on the other side of an inequality reversing its sign.

lemma (in int0) Int˙ZF˙2˙L9B: assumes i∈ℤ m∈ℤ k∈ℤ
shows i+m ≤ k ⟷ i ≤ k-m
⟨*proof*⟩

A special case of Int˙ZF˙2˙L9B with weaker assumptions.

lemma (in int0) Int˙ZF˙2˙L9C:
assumes i∈**ℤ** m∈**ℤ** and i-m ≤ k
shows i ≤ k+m
⟨*proof*⟩

Taking (higher order) minus on both sides of inequality reverses it.

lemma (in int0) Int˙ZF˙2˙L10: assumes k ≤ i
shows
(-i) ≤ (-k)
$-i ≤ $-k
⟨*proof*⟩

Taking minus on both sides of inequality reverses it, version with a negative on one side.

lemma (in int0) Int˙ZF˙2˙L10AA: assumes n∈**ℤ** m≤(-n)
shows n≤(-m)
⟨*proof*⟩

We can cancel the same element on on both sides of an inequality, a version with minus on both sides.

lemma (in int0) Int˙ZF˙2˙L10AB:
assumes m∈**ℤ** n∈**ℤ** k∈**ℤ** and m-n ≤ m-k
shows k≤n
⟨*proof*⟩

If an integer is nonpositive, then its opposite is nonnegative.

lemma (in int0) Int˙ZF˙2˙L10A: assumes k ≤ **0**
shows **0**≤(-k)
⟨*proof*⟩

If the opposite of an integers is nonnegative, then the integer is nonpositive.

lemma (in int0) Int˙ZF˙2˙L10B:
assumes k∈**ℤ** and **0**≤(-k)
shows k≤**0**
⟨*proof*⟩

Adding one to an integer corresponds to taking a successor for a natural number.

lemma (in int0) Int˙ZF˙2˙L11:
shows i $+ $# n $+ ($# 1) = i $+ $# succ(n)
⟨*proof*⟩

Adding a natural number increases integers.

lemma (in int0) Int˙ZF˙2˙L12: assumes A1: i∈**ℤ** and A2: n∈nat
shows i ≤ i $+ $#n
⟨*proof*⟩

Adding one increases integers.

lemma (in int0) Int˙ZF˙2˙L12A: assumes A1: j≤k
  shows j ≤ k $+ $#1 j ≤ k+**1**
⟨*proof*⟩

Adding one increases integers, yet one more version.

lemma (in int0) Int˙ZF˙2˙L12B: assumes A1: m∈ℤ shows m ≤ m+**1**
  ⟨*proof*⟩

If $k + 1 = m + n$, where n is a non-zero natural number, then $m \leq k$.

lemma (in int0) Int˙ZF˙2˙L13:
  assumes A1: k∈ℤ m∈ℤ and A2: n∈nat
  and A3: k $+ ($# 1) = m $+ $# succ(n)
  shows m ≤ k
⟨*proof*⟩

The absolute value of an integer is an integer.

lemma (in int0) Int˙ZF˙2˙L14: assumes A1: m∈ℤ
  shows abs(m) ∈ ℤ
⟨*proof*⟩

If two integers are nonnegative, then the opposite of one is less or equal than the other and the sum is also nonnegative.

lemma (in int0) Int˙ZF˙2˙L14A:
  assumes **0**≤m **0**≤n
  shows
  (-m) ≤ n
  0 ≤ m + n
  ⟨*proof*⟩

We can increase components in an estimate.

lemma (in int0) Int˙ZF˙2˙L15:
  assumes b≤b_1 c≤c_1 and a≤b+c
  shows a≤b_1+c_1
⟨*proof*⟩

We can add or subtract the sides of two inequalities.

lemma (in int0) int˙ineq˙add˙sides:
  assumes a≤b and c≤d
  shows
  a+c ≤ b+d
  a-d ≤ b-c
  ⟨*proof*⟩

We can increase the second component in an estimate.

lemma (in int0) Int˙ZF˙2˙L15A:
  assumes b∈ℤ and a≤b+c and A3: c≤c_1
  shows a≤b+c_1

⟨*proof*⟩

If we increase the second component in a sum of three integers, the whole sum inceases.

lemma (in int0) Int˙ZF˙2˙L15C:
  assumes A1: m∈$\mathbb{Z}$ n∈$\mathbb{Z}$ and A2: k ≤ L
  shows m+k+n ≤ m+L+n
⟨*proof*⟩

We don't decrease an integer by adding a nonnegative one.

lemma (in int0) Int˙ZF˙2˙L15D:
  assumes **0**≤n m∈$\mathbb{Z}$
  shows m ≤ n+m
  ⟨*proof*⟩

Some inequalities about the sum of two integers and its absolute value.

lemma (in int0) Int˙ZF˙2˙L15E:
  assumes m∈$\mathbb{Z}$ n∈$\mathbb{Z}$
  shows
  m+n ≤ abs(m)+abs(n)
  m-n ≤ abs(m)+abs(n)
  (-m)+n ≤ abs(m)+abs(n)
  (-m)-n ≤ abs(m)+abs(n)
  ⟨*proof*⟩

We can add a nonnegative integer to the right hand side of an inequality.

lemma (in int0) Int˙ZF˙2˙L15F: assumes m≤k and **0**≤n
  shows m ≤ k+n m ≤ n+k
  ⟨*proof*⟩

Triangle inequality for integers.

lemma (in int0) Int˙triangle˙ineq:
  assumes m∈$\mathbb{Z}$ n∈$\mathbb{Z}$
  shows abs(m+n)≤abs(m)+abs(n)
  ⟨*proof*⟩

Taking absolute value does not change nonnegative integers.

lemma (in int0) Int˙ZF˙2˙L16:
  assumes **0**≤m shows m∈$\mathbb{Z}^+$ and abs(m) = m
  ⟨*proof*⟩

$0 \leq 1$, so $|1| = 1$.

lemma (in int0) Int˙ZF˙2˙L16A: shows **0**≤**1** and abs(**1**) = **1**
⟨*proof*⟩

$1 \leq 2$.

lemma (in int0) Int˙ZF˙2˙L16B: shows **1**≤**2**

⟨*proof*⟩

Integers greater or equal one are greater or equal zero.

lemma (in int0) Int˙ZF˙2˙L16C:
assumes A1: $\mathbf{1} \leq$a shows
$\mathbf{0} \leq$a a$\neq \mathbf{0}$
$\mathbf{2} \leq$ a+$\mathbf{1}$
$\mathbf{1} \leq$ a+$\mathbf{1}$
$\mathbf{0} \leq$ a+$\mathbf{1}$
⟨*proof*⟩

Absolute value is the same for an integer and its opposite.

lemma (in int0) Int˙ZF˙2˙L17:
assumes m$\in \mathbb{Z}$ shows abs(-m) = abs(m)
⟨*proof*⟩

The absolute value of zero is zero.

lemma (in int0) Int˙ZF˙2˙L18: shows abs($\mathbf{0}$) = $\mathbf{0}$
⟨*proof*⟩

A different version of the triangle inequality.

lemma (in int0) Int˙triangle˙ineq1:
assumes A1: m$\in \mathbb{Z}$ n$\in \mathbb{Z}$
shows
abs(m-n) $\leq$ abs(n)+abs(m)
abs(m-n) $\leq$ abs(m)+abs(n)
⟨*proof*⟩

Another version of the triangle inequality.

lemma (in int0) Int˙triangle˙ineq2:
assumes m$\in \mathbb{Z}$ n$\in \mathbb{Z}$
and abs(m-n) $\leq$ k
shows
abs(m) $\leq$ abs(n)+k
m-k $\leq$ n
m $\leq$ n+k
n-k $\leq$ m
⟨*proof*⟩

Triangle inequality with three integers. We could use OrdGroup˙triangle˙ineq3, but since simp cannot translate the notation directly, it is simpler to reprove it for integers.

lemma (in int0) Int˙triangle˙ineq3:
assumes A1: m$\in \mathbb{Z}$ n$\in \mathbb{Z}$ k$\in \mathbb{Z}$
shows abs(m+n+k) $\leq$ abs(m)+abs(n)+abs(k)
⟨*proof*⟩

The next lemma shows what happens when one integers is not greater or equal than another.

lemma (in int0) Int˙ZF˙2˙L19:
  assumes A1: m∈ℤ n∈ℤ and A2: ¬(n≤m)
  shows m≤n (-n) ≤ (-m) m≠n
⟨*proof*⟩

If one integer is greater or equal and not equal to another, then it is not smaller or equal.

lemma (in int0) Int˙ZF˙2˙L19AA: assumes A1: m≤n and A2: m≠n
  shows ¬(n≤m)
⟨*proof*⟩

The next lemma allows to prove theorems for the case of positive and negative integers separately.

lemma (in int0) Int˙ZF˙2˙L19A: assumes A1: m∈ℤ and A2: ¬(**0**≤m)
  shows m≤**0** **0** ≤ (-m) m≠**0**
⟨*proof*⟩

We can prove a theorem about integers by proving that it holds for $m = 0$, $m \in \mathbb{Z}_+$ and $-m \in \mathbb{Z}_+$.

lemma (in int0) Int˙ZF˙2˙L19B:
  assumes m∈ℤ and Q(**0**) and ∀n∈ℤ₊. Q(n) and ∀n∈ℤ₊. Q(-n)
  shows Q(m)
⟨*proof*⟩

An integer is not greater than its absolute value.

lemma (in int0) Int˙ZF˙2˙L19C: assumes A1: m∈ℤ
  shows
  m ≤ abs(m)
  (-m) ≤ abs(m)
  ⟨*proof*⟩

$|m - n| = |n - m|$.

lemma (in int0) Int˙ZF˙2˙L20: assumes m∈ℤ n∈ℤ
  shows abs(m-n) = abs(n-m)
  ⟨*proof*⟩

We can add the sides of inequalities with absolute values.

lemma (in int0) Int˙ZF˙2˙L21:
  assumes A1: m∈ℤ n∈ℤ
  and A2: abs(m) ≤ k abs(n) ≤ l
  shows
  abs(m+n) ≤ k + l
  abs(m-n) ≤ k + l
  ⟨*proof*⟩

Absolute value is nonnegative.

lemma (in int0) int˙abs˙nonneg: assumes A1: m∈ℤ

shows abs(m) $\in \mathbb{Z}^+$ $\mathbf{0} \leq$ abs(m)
⟨*proof*⟩

If an nonnegative integer is less or equal than another, then so is its absolute value.

lemma (in int0) Int˙ZF˙2˙L23:
assumes $\mathbf{0}\leq$m m$\leq$k
shows abs(m) $\leq$ k
⟨*proof*⟩

42.3 Induction on integers.

In this section we show some induction lemmas for integers. The basic tools are the induction on natural numbers and the fact that integers can be written as a sum of a smaller integer and a natural number.

An integer can be written a a sum of a smaller integer and a natural number.

lemma (in int0) Int˙ZF˙3˙L2: assumes A1: i $\leq$ m
shows $\exists$n$\in$nat. m = i $+ $# n
⟨*proof*⟩

Induction for integers, the induction step.

lemma (in int0) Int˙ZF˙3˙L6: assumes A1: i$\in\mathbb{Z}$
and A2: $\forall$m. i$\leq$m $\wedge$ Q(m) $\longrightarrow$ Q(m $+ ($# 1))
shows $\forall$k$\in$nat. Q(i $+ ($# k)) $\longrightarrow$ Q(i $+ ($# succ(k)))
⟨*proof*⟩

Induction on integers, version with higher-order increment function.

lemma (in int0) Int˙ZF˙3˙L7:
assumes A1: i$\leq$k and A2: Q(i)
and A3: $\forall$m. i$\leq$m $\wedge$ Q(m) $\longrightarrow$ Q(m $+ ($# 1))
shows Q(k)
⟨*proof*⟩

Induction on integer, implication between two forms of the induction step.

lemma (in int0) Int˙ZF˙3˙L7A: assumes
A1: $\forall$m. i$\leq$m $\wedge$ Q(m) $\longrightarrow$ Q(m+$\mathbf{1}$)
shows $\forall$m. i$\leq$m $\wedge$ Q(m) $\longrightarrow$ Q(m $+ ($# 1))
⟨*proof*⟩

Induction on integers, version with ZF increment function.

theorem (in int0) Induction˙on˙int:
assumes A1: i$\leq$k and A2: Q(i)
and A3: $\forall$m. i$\leq$m $\wedge$ Q(m) $\longrightarrow$ Q(m+$\mathbf{1}$)
shows Q(k)
⟨*proof*⟩

Another form of induction on integers. This rewrites the basic theorem Int˙ZF˙3˙L7 substituting $P(-k)$ for $Q(k)$.

lemma (in int0) Int˙ZF˙3˙L7B: assumes A1: i≤k and A2: P($-i)
 and A3: ∀m. i≤m ∧ P($-m) ⟶ P($-(m $+ ($# 1)))
 shows P($-k)
⟨*proof*⟩

Another induction on integers. This rewrites Int_ZF_3_L7 substituting $-k$ for k and $-i$ for i.

lemma (in int0) Int˙ZF˙3˙L8: assumes A1: k≤i and A2: P(i)
 and A3: ∀m. $-i≤m ∧ P($-m) ⟶ P($-(m $+ ($# 1)))
 shows P(k)
⟨*proof*⟩

An implication between two forms of induction steps.

lemma (in int0) Int˙ZF˙3˙L9: assumes A1: i∈**ℤ**
 and A2: ∀n. n≤i ∧ P(n) ⟶ P(n $+ $-($#1))
 shows ∀m. $-i≤m ∧ P($-m) ⟶ P($-(m $+ ($# 1)))
⟨*proof*⟩

Backwards induction on integers, version with higher-order decrement function.

lemma (in int0) Int˙ZF˙3˙L9A: assumes A1: k≤i and A2: P(i)
 and A3: ∀n. n≤i ∧ P(n) ⟶P(n $+ $-($#1))
 shows P(k)
⟨*proof*⟩

Induction on integers, implication between two forms of the induction step.

lemma (in int0) Int˙ZF˙3˙L10: assumes
 A1: ∀n. n≤i ∧ P(n) ⟶ P(n-**1**)
 shows ∀n. n≤i ∧ P(n) ⟶ P(n $+ $-($#1))
⟨*proof*⟩

Backwards induction on integers.

theorem (in int0) Back˙induct˙on˙int:
 assumes A1: k≤i and A2: P(i)
 and A3: ∀n. n≤i ∧ P(n) ⟶ P(n-**1**)
 shows P(k)
⟨*proof*⟩

42.4 Bounded vs. finite subsets of integers

The goal of this section is to establish that a subset of integers is bounded is and only is it is finite. The fact that all finite sets are bounded is already shown for all linearly ordered groups in OrderedGroups˙ZF.thy. To show the other implication we show that all intervals starting at 0 are finite and then use a result from OrderedGroups˙ZF.thy.

There are no integers between k and $k+1$.

lemma (in int0) Int˙ZF˙4˙L1:
  assumes A1: k∈ℤ m∈ℤ n∈nat and A2: k $+ $#1 = m $+ $#n
  shows m = k $+ $#1 ∨ m ≤ k
⟨*proof*⟩

A trivial calculation lemma that allows to subtract and add one.

lemma Int˙ZF˙4˙L1A:
  assumes m∈int shows m $- $#1 $+ $#1 = m
  ⟨*proof*⟩

There are no integers between k and $k+1$, another formulation.

lemma (in int0) Int˙ZF˙4˙L1B: assumes A1: m ≤ L
  shows
  m = L ∨ m+**1** ≤ L
  m = L ∨ m ≤ L-**1**
⟨*proof*⟩

If $j \in m..k+1$, then $j \in m..n$ or $j = k+1$.

lemma (in int0) Int˙ZF˙4˙L2: assumes A1: k∈ℤ
  and A2: j ∈ m..(k $+ $#1)
  shows j ∈ m..k ∨ j ∈ {k $+ $#1}
⟨*proof*⟩

Extending an integer interval by one is the same as adding the new endpoint.

lemma (in int0) Int˙ZF˙4˙L3: assumes A1: m≤ k
  shows m..(k $+ $#1) = m..k ∪ {k $+ $#1}
⟨*proof*⟩

Integer intervals are finite - induction step.

lemma (in int0) Int˙ZF˙4˙L4:
  assumes A1: i≤m and A2: i..m ∈ Fin(ℤ)
  shows i..(m $+ $#1) ∈ Fin(ℤ)
  ⟨*proof*⟩

Integer intervals are finite.

lemma (in int0) Int˙ZF˙4˙L5: assumes A1: i∈ℤ k∈ℤ
  shows i..k ∈ Fin(ℤ)
⟨*proof*⟩

Bounded integer sets are finite.

lemma (in int0) Int˙ZF˙4˙L6: assumes A1: IsBounded(A,IntegerOrder)
  shows A ∈ Fin(ℤ)
⟨*proof*⟩

A subset of integers is bounded iff it is finite.

theorem (in int0) Int˙bounded˙iff˙fin:

shows IsBounded(A,IntegerOrder)⟷ A∈Fin(ℤ)
⟨*proof*⟩

The image of an interval by any integer function is finite, hence bounded.

lemma (in int0) Int˙ZF˙4˙L8:
assumes A1: i∈ℤ k∈ℤ and A2: f:ℤ→ℤ
shows
f(i..k) ∈ Fin(ℤ)
IsBounded(f(i..k),IntegerOrder)
⟨*proof*⟩

If for every integer we can find one in A that is greater or equal, then A is is not bounded above, hence infinite.

lemma (in int0) Int˙ZF˙4˙L9: assumes A1: ∀m∈ℤ. ∃k∈A. m≤k
shows
¬IsBoundedAbove(A,IntegerOrder)
A ∉ Fin(ℤ)
⟨*proof*⟩

end

43 Integers 1

theory Int˙ZF˙1 imports Int˙ZF˙IML OrderedRing˙ZF

begin

This theory file considers the set of integers as an ordered ring.

43.1 Integers as a ring

In this section we show that integers form a commutative ring.

The next lemma provides the condition to show that addition is distributive with respect to multiplication.

lemma (in int0) Int˙ZF˙1˙1˙L1: assumes A1: a∈ℤ b∈ℤ c∈ℤ
shows
a·(b+c) = a·b + a·c
(b+c)·a = b·a + c·a
⟨*proof*⟩

Integers form a commutative ring, hence we can use theorems proven in ring0 context (locale).

lemma (in int0) Int˙ZF˙1˙1˙L2: shows
IsAring(ℤ,IntegerAddition,IntegerMultiplication)
IntegerMultiplication –is commutative on″ ℤ

ring0($\mathbb{Z}$,IntegerAddition,IntegerMultiplication)
$\langle proof \rangle$

Zero and one are integers.

lemma (in int0) int˙zero˙one˙are˙int: shows $\mathbf{0}\in\mathbb{Z}$ $\mathbf{1}\in\mathbb{Z}$
$\langle proof \rangle$

Negative of zero is zero.

lemma (in int0) int˙zero˙one˙are˙intA: shows $(-\mathbf{0}) = \mathbf{0}$
$\langle proof \rangle$

Properties with one integer.

lemma (in int0) Int˙ZF˙1˙1˙L4: assumes A1: $a \in \mathbb{Z}$
shows
$a+\mathbf{0} = a$
$\mathbf{0}+a = a$
$a\cdot\mathbf{1} = a$ $\mathbf{1}\cdot a = a$
$\mathbf{0}\cdot a = \mathbf{0}$ $a\cdot\mathbf{0} = \mathbf{0}$
$(-a) \in \mathbb{Z}$ $(-(-a)) = a$
$a-a = \mathbf{0}$ $a-\mathbf{0} = a$ $\mathbf{2}\cdot a = a+a$
$\langle proof \rangle$

Properties that require two integers.

lemma (in int0) Int˙ZF˙1˙1˙L5: assumes $a\in\mathbb{Z}$ $b\in\mathbb{Z}$
shows
$a+b \in \mathbb{Z}$
$a-b \in \mathbb{Z}$
$a\cdot b \in \mathbb{Z}$
$a+b = b+a$
$a\cdot b = b\cdot a$
$(-b)-a = (-a)-b$
$(-(a+b)) = (-a)-b$
$(-(a-b)) = ((-a)+b)$
$(-a)\cdot b = -(a\cdot b)$
$a\cdot(-b) = -(a\cdot b)$
$(-a)\cdot(-b) = a\cdot b$
$\langle proof \rangle$

2 and 3 are integers.

lemma (in int0) int˙two˙three˙are˙int: shows $\mathbf{2} \in \mathbb{Z}$ $\mathbf{3} \in \mathbb{Z}$
$\langle proof \rangle$

Another property with two integers.

lemma (in int0) Int˙ZF˙1˙1˙L5B:
assumes $a\in\mathbb{Z}$ $b\in\mathbb{Z}$
shows $a-(-b) = a+b$
$\langle proof \rangle$

Properties that require three integers.

lemma (in int0) Int˙ZF˙1˙1˙L6: assumes a∈$\mathbb{Z}$ b∈$\mathbb{Z}$ c∈$\mathbb{Z}$
shows
a-(b+c) = a-b-c
a-(b-c) = a-b+c
a·(b-c) = a·b - a·c
(b-c)·a = b·a - c·a
⟨*proof*⟩

One more property with three integers.

lemma (in int0) Int˙ZF˙1˙1˙L6A: assumes a∈$\mathbb{Z}$ b∈$\mathbb{Z}$ c∈$\mathbb{Z}$
shows a+(b-c) = a+b-c
⟨*proof*⟩

Associativity of addition and multiplication.

lemma (in int0) Int˙ZF˙1˙1˙L7: assumes a∈$\mathbb{Z}$ b∈$\mathbb{Z}$ c∈$\mathbb{Z}$
shows
a+b+c = a+(b+c)
a·b·c = a·(b·c)
⟨*proof*⟩

43.2 Rearrangement lemmas

In this section we collect lemmas about identities related to rearranging the terms in expresssions

A formula with a positive integer.

lemma (in int0) Int˙ZF˙1˙2˙L1: assumes **0**≤a
shows abs(a)+**1** = abs(a+**1**)
⟨*proof*⟩

A formula with two integers, one positive.

lemma (in int0) Int˙ZF˙1˙2˙L2: assumes A1: a∈$\mathbb{Z}$ and A2: **0**≤b
shows a+(abs(b)+**1**)·a = (abs(b+**1**)+**1**)·a
⟨*proof*⟩

A couple of formulae about canceling opposite integers.

lemma (in int0) Int˙ZF˙1˙2˙L3: assumes A1: a∈$\mathbb{Z}$ b∈$\mathbb{Z}$
shows
a+b-a = b
a+(b-a) = b
a+b-b = a
a-b+b = a
(-a)+(a+b) = b
a+(b-a) = b
(-b)+(a+b) = a
a-(b+a) = -b

a-(a+b) = -b
a-(a-b) = b
a-b-a = -b
a-b - (a+b) = (-b)-b
⟨*proof*⟩

Subtracting one does not increase integers. This may be moved to a theory about ordered rings one day.

lemma (in int0) Int_ZF_1_2_L3A: assumes A1: a≤b
shows a-**1** ≤ b
⟨*proof*⟩

Subtracting one does not increase integers, special case.

lemma (in int0) Int_ZF_1_2_L3AA:
assumes A1: a∈ℤ shows
a-**1** ≤a
a-**1** ≠ a
¬(a≤a-**1**)
¬(a+**1** ≤a)
¬(**1**+a ≤a)
⟨*proof*⟩

A formula with a nonpositive integer.

lemma (in int0) Int_ZF_1_2_L4: assumes a≤**0**
shows abs(a)+**1** = abs(a-**1**)
⟨*proof*⟩

A formula with two integers, one negative.

lemma (in int0) Int_ZF_1_2_L5: assumes A1: a∈ℤ and A2: b≤**0**
shows a+(abs(b)+**1**)·a = (abs(b-**1**)+**1**)·a
⟨*proof*⟩

A rearrangement with four integers.

lemma (in int0) Int_ZF_1_2_L6:
assumes A1: a∈ℤ b∈ℤ c∈ℤ d∈ℤ
shows
a-(b-**1**)·c = (d-b·c)-(d-a-c)
⟨*proof*⟩

Some other rearrangements with two integers.

lemma (in int0) Int_ZF_1_2_L7: assumes a∈ℤ b∈ℤ
shows
a·b = (a-**1**)·b+b
a·(b+**1**) = a·b+a
(b+**1**)·a = b·a+a
(b+**1**)·a = a+b·a
⟨*proof*⟩

Another rearrangement with two integers.

lemma (in int0) Int˙ZF˙1˙2˙L8:
  assumes A1: a∈ℤ b∈ℤ
  shows a+**1**+(b+**1**) = b+a+**2**
  ⟨*proof*⟩

A couple of rearrangement with three integers.

lemma (in int0) Int˙ZF˙1˙2˙L9:
  assumes a∈ℤ b∈ℤ c∈ℤ
  shows
  (a-b)+(b-c) = a-c
  (a-b)-(a-c) = c-b
  a+(b+(c-a-b)) = c
  (-a)-b+c = c-a-b
  (-b)-a+c = c-a-b
  (-((-a)+b+c)) = a-b-c
  a+b+c-a = b+c
  a+b-(a+c) = b-c
  ⟨*proof*⟩

Another couple of rearrangements with three integers.

lemma (in int0) Int˙ZF˙1˙2˙L9A:
  assumes A1: a∈ℤ b∈ℤ c∈ℤ
  shows (-(a-b-c)) = c+b-a
⟨*proof*⟩

Another rearrangement with three integers.

lemma (in int0) Int˙ZF˙1˙2˙L10:
  assumes A1: a∈ℤ b∈ℤ c∈ℤ
  shows (a+**1**)·b + (c+**1**)·b = (c+a+**2**)·b
⟨*proof*⟩

A technical rearrangement involing inequalities with absolute value.

lemma (in int0) Int˙ZF˙1˙2˙L10A:
  assumes A1: a∈ℤ b∈ℤ c∈ℤ e∈ℤ
  and A2: abs(a·b-c) ≤ d abs(b·a-e) ≤ f
  shows abs(c-e) ≤ f+d
⟨*proof*⟩

Some arithmetics.

lemma (in int0) Int˙ZF˙1˙2˙L11: assumes A1: a∈ℤ
  shows
  a+**1**+**2** = a+**3**
  a = **2**·a - a
⟨*proof*⟩

A simple rearrangement with three integers.

lemma (in int0) Int˙ZF˙1˙2˙L12:

assumes a∈$\mathbb{Z}$ b∈$\mathbb{Z}$ c∈$\mathbb{Z}$
shows
(b-c)·a = a·b - a·c
⟨*proof*⟩

A big rearrangement with five integers.

lemma (in int0) Int˙ZF˙1˙2˙L13:
assumes A1: a∈$\mathbb{Z}$ b∈$\mathbb{Z}$ c∈$\mathbb{Z}$ d∈$\mathbb{Z}$ x∈$\mathbb{Z}$
shows (x+(a·x+b)+c)·d = d·(a+**1**)·x + (b·d+c·d)
⟨*proof*⟩

Rerrangement about adding linear functions.

lemma (in int0) Int˙ZF˙1˙2˙L14:
assumes a∈$\mathbb{Z}$ b∈$\mathbb{Z}$ c∈$\mathbb{Z}$ d∈$\mathbb{Z}$ x∈$\mathbb{Z}$
shows (a·x + b) + (c·x + d) = (a+c)·x + (b+d)
⟨*proof*⟩

A rearrangement with four integers. Again we have to use the generic set notation to use a theorem proven in different context.

lemma (in int0) Int˙ZF˙1˙2˙L15: assumes A1: a∈$\mathbb{Z}$ b∈$\mathbb{Z}$ c∈$\mathbb{Z}$ d∈$\mathbb{Z}$
and A2: a = b-c-d
shows
d = b-a-c
d = (-a)+b-c
b = a+d+c
⟨*proof*⟩

A rearrangement with four integers. Property of groups.

lemma (in int0) Int˙ZF˙1˙2˙L16:
assumes a∈$\mathbb{Z}$ b∈$\mathbb{Z}$ c∈$\mathbb{Z}$ d∈$\mathbb{Z}$
shows a+(b-c)+d = a+b+d-c
⟨*proof*⟩

Some rearrangements with three integers. Properties of groups.

lemma (in int0) Int˙ZF˙1˙2˙L17:
assumes A1: a∈$\mathbb{Z}$ b∈$\mathbb{Z}$ c∈$\mathbb{Z}$
shows
a+b-c+(c-b) = a
a+(b+c)-c = a+b
⟨*proof*⟩

Another rearrangement with three integers. Property of abelian groups.

lemma (in int0) Int˙ZF˙1˙2˙L18:
assumes A1: a∈$\mathbb{Z}$ b∈$\mathbb{Z}$ c∈$\mathbb{Z}$
shows a+b-c+(c-a) = b
⟨*proof*⟩

43.3 Integers as an ordered ring

We already know from Int˙ZF that integers with addition form a linearly ordered group. To show that integers form an ordered ring we need the fact that the set of nonnegative integers is closed under multiplication.

We start with the property that a product of nonnegative integers is nonnegative. The proof is by induction and the next lemma is the induction step.

lemma (in int0) Int˙ZF˙1˙3˙L1: assumes A1: $\mathbf{0}$≤a $\mathbf{0}$≤b
  and A3: $\mathbf{0}$ ≤ a·b
  shows $\mathbf{0}$ ≤ a·(b+$\mathbf{1}$)
⟨*proof*⟩

Product of nonnegative integers is nonnegative.

lemma (in int0) Int˙ZF˙1˙3˙L2: assumes A1: $\mathbf{0}$≤a $\mathbf{0}$≤b
  shows $\mathbf{0}$≤a·b
⟨*proof*⟩

The set of nonnegative integers is closed under multiplication.

lemma (in int0) Int˙ZF˙1˙3˙L2A: shows
  $\mathbb{Z}^+$ –is closed under″ IntegerMultiplication
⟨*proof*⟩

Integers form an ordered ring. All theorems proven in the ring1 context are valid in int0 context.

theorem (in int0) Int˙ZF˙1˙3˙T1: shows
  IsAnOrdRing($\mathbb{Z}$,IntegerAddition,IntegerMultiplication,IntegerOrder)
  ring1($\mathbb{Z}$,IntegerAddition,IntegerMultiplication,IntegerOrder)
  ⟨*proof*⟩

Product of integers that are greater that one is greater than one. The proof is by induction and the next step is the induction step.

lemma (in int0) Int˙ZF˙1˙3˙L3˙indstep:
  assumes A1: $\mathbf{1}$≤a $\mathbf{1}$≤b
  and A2: $\mathbf{1}$ ≤ a·b
  shows $\mathbf{1}$ ≤ a·(b+$\mathbf{1}$)
⟨*proof*⟩

Product of integers that are greater that one is greater than one.

lemma (in int0) Int˙ZF˙1˙3˙L3:
  assumes A1: $\mathbf{1}$≤a $\mathbf{1}$≤b
  shows $\mathbf{1}$ ≤ a·b
⟨*proof*⟩

$|a \cdot (-b)| = |(-a) \cdot b| = |(-a) \cdot (-b)| = |a \cdot b|$ This is a property of ordered rings..

lemma (in int0) Int_ZF_1_3_L4: assumes a∈ℤ b∈ℤ
  shows
  abs((-a)·b) = abs(a·b)
  abs(a·(-b)) = abs(a·b)
  abs((-a)·(-b)) = abs(a·b)
  ⟨*proof*⟩

Absolute value of a product is the product of absolute values. Property of ordered rings.

lemma (in int0) Int_ZF_1_3_L5:
  assumes A1: a∈ℤ b∈ℤ
  shows abs(a·b) = abs(a)·abs(b)
  ⟨*proof*⟩

Double nonnegative is nonnegative. Property of ordered rings.

lemma (in int0) Int_ZF_1_3_L5A: assumes **0**≤a
  shows **0**≤**2**·a
  ⟨*proof*⟩

The next lemma shows what happens when one integer is not greater or equal than another.

lemma (in int0) Int_ZF_1_3_L6:
  assumes A1: a∈ℤ b∈ℤ
  shows ¬(b≤a) ⟷ a+**1** ≤ b
⟨*proof*⟩

Another form of stating that there are no integers between integers m and $m + 1$.

corollary (in int0) no_int_between: assumes A1: a∈ℤ b∈ℤ
  shows b≤a ∨ a+**1** ≤ b
  ⟨*proof*⟩

Another way of saying what it means that one integer is not greater or equal than another.

corollary (in int0) Int_ZF_1_3_L6A:
  assumes A1: a∈ℤ b∈ℤ and A2: ¬(b≤a)
  shows a ≤ b-**1**
⟨*proof*⟩

Yet another form of stating that there are nointegers between m and $m+1$.

lemma (in int0) no_int_between1:
  assumes A1: a≤b and A2: a≠b
  shows
  a+**1** ≤ b
  a ≤ b-**1**
⟨*proof*⟩

We can decompose proofs into three cases: $a = b$, $a \leq b - 1b$ or $a \geq b + 1b$.

lemma (in int0) Int_ZF_1_3_L6B: assumes A1: a∈ℤ b∈ℤ
  shows a=b ∨ (a ≤ b-**1**) ∨ (b+**1** ≤a)
⟨*proof*⟩

A special case of Int_ZF_1_3_L6B when $b = 0$. This allows to split the proofs in cases $a \leq -1$, $a = 0$ and $a \geq 1$.

corollary (in int0) Int_ZF_1_3_L6C: assumes A1: a∈ℤ
  shows a=**0** ∨ (a ≤ -**1**) ∨ (**1**≤a)
⟨*proof*⟩

An integer is not less or equal zero iff it is greater or equal one.

lemma (in int0) Int_ZF_1_3_L7: assumes a∈ℤ
  shows ¬(a≤**0**) ⟷ **1** ≤ a
  ⟨*proof*⟩

Product of positive integers is positive.

lemma (in int0) Int_ZF_1_3_L8:
  assumes a∈ℤ b∈ℤ
  and ¬(a≤**0**) ¬(b≤**0**)
  shows ¬((a·b) ≤ **0**)
  ⟨*proof*⟩

If $a \cdot b$ is nonnegative and b is positive, then a is nonnegative. Proof by contradiction.

lemma (in int0) Int_ZF_1_3_L9:
  assumes A1: a∈ℤ b∈ℤ
  and A2: ¬(b≤**0**) and A3: a·b ≤ **0**
  shows a≤**0**
⟨*proof*⟩

One integer is less or equal another iff the difference is nonpositive.

lemma (in int0) Int_ZF_1_3_L10:
  assumes a∈ℤ b∈ℤ
  shows a≤b ⟷ a-b ≤ **0**
  ⟨*proof*⟩

Some conclusions from the fact that one integer is less or equal than another.

lemma (in int0) Int_ZF_1_3_L10A: assumes a≤b
  shows **0** ≤ b-a
  ⟨*proof*⟩

We can simplify out a positive element on both sides of an inequality.

lemma (in int0) Int_ineq_simpl_positive:
  assumes A1: a∈ℤ b∈ℤ c∈ℤ
  and A2: a·c ≤ b·c and A4: ¬(c≤**0**)
  shows a ≤ b
⟨*proof*⟩

A technical lemma about conclusion from an inequality between absolute values. This is a property of ordered rings.

lemma (in int0) Int_ZF_1_3_L11:
  assumes A1: $a \in \mathbb{Z}$ $b \in \mathbb{Z}$
  and A2: $\neg(\text{abs}(a) \leq \text{abs}(b))$
  shows $\neg(\text{abs}(a) \leq \mathbf{0})$
⟨*proof*⟩

Negative times positive is negative. This a property of ordered rings.

lemma (in int0) Int_ZF_1_3_L12:
  assumes $a \leq \mathbf{0}$ and $\mathbf{0} \leq b$
  shows $a \cdot b \leq \mathbf{0}$
  ⟨*proof*⟩

We can multiply an inequality by a nonnegative number. This is a property of ordered rings.

lemma (in int0) Int_ZF_1_3_L13:
  assumes A1: $a \leq b$ and A2: $\mathbf{0} \leq c$
  shows
  $a \cdot c \leq b \cdot c$
  $c \cdot a \leq c \cdot b$
  ⟨*proof*⟩

A technical lemma about decreasing a factor in an inequality.

lemma (in int0) Int_ZF_1_3_L13A:
  assumes $\mathbf{1} \leq a$ and $b \leq c$ and $(a+\mathbf{1}) \cdot c \leq d$
  shows $(a+\mathbf{1}) \cdot b \leq d$
⟨*proof*⟩

We can multiply an inequality by a positive number. This is a property of ordered rings.

lemma (in int0) Int_ZF_1_3_L13B:
  assumes A1: $a \leq b$ and A2: $c \in \mathbb{Z}_+$
  shows
  $a \cdot c \leq b \cdot c$
  $c \cdot a \leq c \cdot b$
⟨*proof*⟩

A rearrangement with four integers and absolute value.

lemma (in int0) Int_ZF_1_3_L14:
  assumes A1: $a \in \mathbb{Z}$ $b \in \mathbb{Z}$ $c \in \mathbb{Z}$ $d \in \mathbb{Z}$
  shows $\text{abs}(a \cdot b)+(\text{abs}(a)+c) \cdot d = (d+\text{abs}(b)) \cdot \text{abs}(a)+c \cdot d$
⟨*proof*⟩

A technical lemma about what happens when one absolute value is not greater or equal than another.

lemma (in int0) Int_ZF_1_3_L15: assumes A1: $m \in \mathbb{Z}$ $n \in \mathbb{Z}$

and A2: $\neg$(abs(m) $\leq$ abs(n))
shows n $\leq$ abs(m) m$\neq$**0**
⟨*proof*⟩

Negative of a nonnegative is nonpositive.

lemma (in int0) Int˙ZF˙1˙3˙L16: assumes A1: **0** $\leq$ m
shows (-m) $\leq$ **0**
⟨*proof*⟩

Some statements about intervals centered at 0.

lemma (in int0) Int˙ZF˙1˙3˙L17: assumes A1: m$\in\mathbb{Z}$
shows
(-abs(m)) $\leq$ abs(m)
(-abs(m))..abs(m) $\neq$ 0
⟨*proof*⟩

The greater of two integers is indeed greater than both, and the smaller one is smaller that both.

lemma (in int0) Int˙ZF˙1˙3˙L18: assumes A1: m$\in\mathbb{Z}$ n$\in\mathbb{Z}$
shows
m $\leq$ GreaterOf(IntegerOrder,m,n)
n $\leq$ GreaterOf(IntegerOrder,m,n)
SmallerOf(IntegerOrder,m,n) $\leq$ m
SmallerOf(IntegerOrder,m,n) $\leq$ n
⟨*proof*⟩

If $|m| \leq n$, then $m \in -n..n$.

lemma (in int0) Int˙ZF˙1˙3˙L19:
assumes A1: m$\in\mathbb{Z}$ and A2: abs(m) $\leq$ n
shows
(-n) $\leq$ m m $\leq$ n
m $\in$ (-n)..n
0 $\leq$ n
⟨*proof*⟩

A slight generalization of the above lemma.

lemma (in int0) Int˙ZF˙1˙3˙L19A:
assumes A1: m$\in\mathbb{Z}$ and A2: abs(m) $\leq$ n and A3: **0**$\leq$k
shows (-(n+k)) $\leq$ m
⟨*proof*⟩

Sets of integers that have absolute value bounded are bounded.

lemma (in int0) Int˙ZF˙1˙3˙L20:
assumes A1: $\forall$x$\in$X. b(x) $\in$ $\mathbb{Z}$ $\wedge$ abs(b(x)) $\leq$ L
shows IsBounded(–b(x). x$\in$X″,IntegerOrder)
⟨*proof*⟩

If a set is bounded, then the absolute values of the elements of that set are bounded.

lemma (in int0) Int_ZF_1_3_L20A: assumes IsBounded(A,IntegerOrder)
  shows ∃L. ∀a∈A. abs(a) ≤ L
  ⟨*proof*⟩

Absolute vaues of integers from a finite image of integers are bounded by an integer.

lemma (in int0) Int_ZF_1_3_L20AA:
  assumes A1: {b(x). x∈ℤ} ∈ Fin(ℤ)
  shows ∃L∈ℤ. ∀x∈ℤ. abs(b(x)) ≤ L
  ⟨*proof*⟩

If absolute values of values of some integer function are bounded, then the image a set from the domain is a bounded set.

lemma (in int0) Int_ZF_1_3_L20B:
  assumes f:X→ℤ and A⊆X and ∀x∈A. abs(f(x)) ≤ L
  shows IsBounded(f(A),IntegerOrder)
⟨*proof*⟩

A special case of the previous lemma for a function from integers to integers.

corollary (in int0) Int_ZF_1_3_L20C:
  assumes f:ℤ→ℤ and ∀m∈ℤ. abs(f(m)) ≤ L
  shows f(ℤ) ∈ Fin(ℤ)
⟨*proof*⟩

A triangle inequality with three integers. Property of linearly ordered abelian groups.

lemma (in int0) int_triangle_ineq3:
  assumes A1: a∈ℤ b∈ℤ c∈ℤ
  shows abs(a-b-c) ≤ abs(a) + abs(b) + abs(c)
⟨*proof*⟩

If $a \leq c$ and $b \leq c$, then $a + b \leq 2 \cdot c$. Property of ordered rings.

lemma (in int0) Int_ZF_1_3_L21:
  assumes A1: a≤c b≤c shows a+b ≤ **2**·c
  ⟨*proof*⟩

If an integer a is between b and $b + c$, then $|b - a| \leq c$. Property of ordered groups.

lemma (in int0) Int_ZF_1_3_L22:
  assumes a≤b and c∈ℤ and b≤ c+a
  shows abs(b-a) ≤ c
  ⟨*proof*⟩

An application of the triangle inequality with four integers. Property of linearly ordered abelian groups.

lemma (in int0) Int_ZF_1_3_L22A:
  assumes a∈ℤ b∈ℤ c∈ℤ d∈ℤ

shows abs(a-c) $\leq$ abs(a+b) + abs(c+d) + abs(b-d)
⟨*proof*⟩

If an integer a is between b and $b+c$, then $|b-a| \leq c$. Property of ordered groups. A version of Int˙ZF˙1˙3˙L22 with sligtly different assumptions.

lemma (in int0) Int˙ZF˙1˙3˙L23:
assumes A1: a$\leq$b and A2: c$\in\mathbb{Z}$ and A3: b$\leq$ a+c
shows abs(b-a) $\leq$ c
⟨*proof*⟩

43.4 Maximum and minimum of a set of integers

In this section we provide some sufficient conditions for integer subsets to have extrema (maxima and minima).

Finite nonempty subsets of integers attain maxima and minima.

theorem (in int0) Int˙fin˙have˙max˙min:
assumes A1: A $\in$ Fin($\mathbb{Z}$) and A2: A$\neq$0
shows
HasAmaximum(IntegerOrder,A)
HasAminimum(IntegerOrder,A)
Maximum(IntegerOrder,A) $\in$ A
Minimum(IntegerOrder,A) $\in$ A
$\forall$x$\in$A. x $\leq$ Maximum(IntegerOrder,A)
$\forall$x$\in$A. Minimum(IntegerOrder,A) $\leq$ x
Maximum(IntegerOrder,A) $\in \mathbb{Z}$
Minimum(IntegerOrder,A) $\in \mathbb{Z}$
⟨*proof*⟩

Bounded nonempty integer subsets attain maximum and minimum.

theorem (in int0) Int˙bounded˙have˙max˙min:
assumes IsBounded(A,IntegerOrder) and A$\neq$0
shows
HasAmaximum(IntegerOrder,A)
HasAminimum(IntegerOrder,A)
Maximum(IntegerOrder,A) $\in$ A
Minimum(IntegerOrder,A) $\in$ A
$\forall$x$\in$A. x $\leq$ Maximum(IntegerOrder,A)
$\forall$x$\in$A. Minimum(IntegerOrder,A) $\leq$ x
Maximum(IntegerOrder,A) $\in \mathbb{Z}$
Minimum(IntegerOrder,A) $\in \mathbb{Z}$
⟨*proof*⟩

Nonempty set of integers that is bounded below attains its minimum.

theorem (in int0) int˙bounded˙below˙has˙min:
assumes A1: IsBoundedBelow(A,IntegerOrder) and A2: A$\neq$0
shows
HasAminimum(IntegerOrder,A)

Minimum(IntegerOrder,A) $\in$ A

$\forall$x$\in$A. Minimum(IntegerOrder,A) $\leq$ x
⟨*proof*⟩

Nonempty set of integers that is bounded above attains its maximum.

theorem (in int0) int˙bounded˙above˙has˙max:
assumes A1: IsBoundedAbove(A,IntegerOrder) and A2: A$\neq$0
shows
HasAmaximum(IntegerOrder,A)
Maximum(IntegerOrder,A) $\in$ A
Maximum(IntegerOrder,A) $\in \mathbb{Z}$
$\forall$x$\in$A. x $\leq$ Maximum(IntegerOrder,A)
⟨*proof*⟩

A set defined by separation over a bounded set attains its maximum and minimum.

lemma (in int0) Int˙ZF˙1˙4˙L1:
assumes A1: IsBounded(A,IntegerOrder) and A2: A$\neq$0
and A3: $\forall$q$\in\mathbb{Z}$. F(q) $\in \mathbb{Z}$
and A4: K = –F(q). q $\in$ A″
shows
HasAmaximum(IntegerOrder,K)
HasAminimum(IntegerOrder,K)
Maximum(IntegerOrder,K) $\in$ K
Minimum(IntegerOrder,K) $\in$ K
Maximum(IntegerOrder,K) $\in \mathbb{Z}$
Minimum(IntegerOrder,K) $\in \mathbb{Z}$
$\forall$q$\in$A. F(q) $\leq$ Maximum(IntegerOrder,K)
$\forall$q$\in$A. Minimum(IntegerOrder,K) $\leq$ F(q)
IsBounded(K,IntegerOrder)
⟨*proof*⟩

A three element set has a maximume and minimum.

lemma (in int0) Int˙ZF˙1˙4˙L1A: assumes A1: a$\in\mathbb{Z}$ b$\in\mathbb{Z}$ c$\in\mathbb{Z}$
shows
Maximum(IntegerOrder,–a,b,c″) $\in \mathbb{Z}$
a $\leq$ Maximum(IntegerOrder,–a,b,c″)
b $\leq$ Maximum(IntegerOrder,–a,b,c″)
c $\leq$ Maximum(IntegerOrder,–a,b,c″)
⟨*proof*⟩

Integer functions attain maxima and minima over intervals.

lemma (in int0) Int˙ZF˙1˙4˙L2:
assumes A1: f:$\mathbb{Z}\rightarrow\mathbb{Z}$ and A2: a$\leq$b
shows
maxf(f,a..b) $\in \mathbb{Z}$
$\forall$c $\in$ a..b. f(c) $\leq$ maxf(f,a..b)

$\exists c \in a..b.\ f(c) = maxf(f,a..b)$
$minf(f,a..b) \in \mathbb{Z}$
$\forall c \in a..b.\ minf(f,a..b) \leq f(c)$
$\exists c \in a..b.\ f(c) = minf(f,a..b)$
⟨*proof*⟩

43.5 The set of nonnegative integers

The set of nonnegative integers looks like the set of natural numbers. We explore that in this section. We also rephrase some lemmas about the set of positive integers known from the theory of oredered grups.

The set of positive integers is closed under addition.

lemma (in int0) pos˙int˙closed˙add:
shows $\mathbb{Z}_+$ –is closed under″ IntegerAddition
⟨*proof*⟩

Text expended version of the fact that the set of positive integers is closed under addition

lemma (in int0) pos˙int˙closed˙add˙unfolded:
assumes $a\in\mathbb{Z}_+$ $b\in\mathbb{Z}_+$ shows $a+b \in \mathbb{Z}_+$
⟨*proof*⟩

$\mathbb{Z}^+$ is bounded below.

lemma (in int0) Int˙ZF˙1˙5˙L1: shows
IsBoundedBelow($\mathbb{Z}^+$,IntegerOrder)
IsBoundedBelow($\mathbb{Z}_+$,IntegerOrder)
⟨*proof*⟩

Subsets of $\mathbb{Z}^+$ are bounded below.

lemma (in int0) Int˙ZF˙1˙5˙L1A: assumes $A \subseteq \mathbb{Z}^+$
shows IsBoundedBelow(A,IntegerOrder)
⟨*proof*⟩

Subsets of $\mathbb{Z}_+$ are bounded below.

lemma (in int0) Int˙ZF˙1˙5˙L1B: assumes A1: $A \subseteq \mathbb{Z}_+$
shows IsBoundedBelow(A,IntegerOrder)
⟨*proof*⟩

Every nonempty subset of positive integers has a mimimum.

lemma (in int0) Int˙ZF˙1˙5˙L1C: assumes $A \subseteq \mathbb{Z}_+$ and $A \neq 0$
shows
HasAminimum(IntegerOrder,A)
Minimum(IntegerOrder,A) $\in$ A
$\forall x\in A.$ Minimum(IntegerOrder,A) $\leq$ x
⟨*proof*⟩

Infinite subsets of Z^+ do not have a maximum - If $A \subseteq Z^+$ then for every integer we can find one in the set that is not smaller.

lemma (in int0) Int˙ZF˙1˙5˙L2:
  assumes A1: A ⊆ $\mathbb{Z}^+$ and A2: A ∉ Fin($\mathbb{Z}$) and A3: D∈$\mathbb{Z}$
  shows ∃n∈A. D≤n
⟨*proof*⟩

Infinite subsets of Z_+ do not have a maximum - If $A \subseteq Z_+$ then for every integer we can find one in the set that is not smaller. This is very similar to Int˙ZF˙1˙5˙L2, except we have $\mathbb{Z}_+$ instead of $\mathbb{Z}^+$ here.

lemma (in int0) Int˙ZF˙1˙5˙L2A:
  assumes A1: A ⊆ $\mathbb{Z}_+$ and A2: A ∉ Fin($\mathbb{Z}$) and A3: D∈$\mathbb{Z}$
  shows ∃n∈A. D≤n
⟨*proof*⟩

An integer is either positive, zero, or its opposite is postitive.

lemma (in int0) Int˙decomp: assumes m∈$\mathbb{Z}$
  shows Exactly˙1˙of˙3˙holds (m=**0**,m∈$\mathbb{Z}_+$,(-m)∈$\mathbb{Z}_+$)
  ⟨*proof*⟩

An integer is zero, positive, or it's inverse is positive.

lemma (in int0) int˙decomp˙cases: assumes m∈$\mathbb{Z}$
  shows m=**0** ∨ m∈$\mathbb{Z}_+$ ∨ (-m) ∈ $\mathbb{Z}_+$
  ⟨*proof*⟩

An integer is in the positive set iff it is greater or equal one.

lemma (in int0) Int˙ZF˙1˙5˙L3: shows m∈$\mathbb{Z}_+$ ⟷ **1**≤m
⟨*proof*⟩

The set of positive integers is closed under multiplication. The unfolded form.

lemma (in int0) pos˙int˙closed˙mul˙unfold:
  assumes a∈$\mathbb{Z}_+$ b∈$\mathbb{Z}_+$
  shows a·b ∈ $\mathbb{Z}_+$
  ⟨*proof*⟩

The set of positive integers is closed under multiplication.

lemma (in int0) pos˙int˙closed˙mul: shows
  $\mathbb{Z}_+$ –is closed under˝ IntegerMultiplication
  ⟨*proof*⟩

It is an overkill to prove that the ring of integers has no zero divisors this way, but why not?

lemma (in int0) int˙has˙no˙zero˙divs:
  shows HasNoZeroDivs($\mathbb{Z}$,IntegerAddition,IntegerMultiplication)
  ⟨*proof*⟩

Nonnegative integers are positive ones plus zero.

lemma (in int0) Int_ZF_1_5_L3A: shows $\mathbb{Z}^+ = \mathbb{Z}_+ \cup \{\mathbf{0}\}$
⟨*proof*⟩

We can make a function smaller than any constant on a given interval of positive integers by adding another constant.

lemma (in int0) Int_ZF_1_5_L4:
assumes A1: f:$\mathbb{Z}\rightarrow\mathbb{Z}$ and A2: K$\in\mathbb{Z}$ N$\in\mathbb{Z}$
shows $\exists$ C$\in\mathbb{Z}$. $\forall$ n$\in\mathbb{Z}_+$. K $\leq$ f(n) + C $\longrightarrow$ N$\leq$n
⟨*proof*⟩

Absolute value is identity on positive integers.

lemma (in int0) Int_ZF_1_5_L4A:
assumes a$\in\mathbb{Z}_+$ shows abs(a) = a
⟨*proof*⟩

One and two are in $\mathbb{Z}_+$.

lemma (in int0) int_one_two_are_pos: shows $\mathbf{1} \in \mathbb{Z}_+$ $\mathbf{2} \in \mathbb{Z}_+$
⟨*proof*⟩

The image of $\mathbb{Z}_+$ by a function defined on integers is not empty.

lemma (in int0) Int_ZF_1_5_L5: assumes A1: f : $\mathbb{Z}\rightarrow$X
shows f($\mathbb{Z}_+$) $\neq$ 0
⟨*proof*⟩

If n is positive, then $n-1$ is nonnegative.

lemma (in int0) Int_ZF_1_5_L6: assumes A1: n $\in \mathbb{Z}_+$
shows
$\mathbf{0} \leq$ n-$\mathbf{1}$
$\mathbf{0} \in \mathbf{0}$..(n-$\mathbf{1}$)
$\mathbf{0}$..(n-$\mathbf{1}$) $\subseteq \mathbb{Z}$
⟨*proof*⟩

Intgers greater than one in $\mathbb{Z}_+$ belong to $\mathbb{Z}_+$. This is a property of ordered groups and follows from OrderedGroup_ZF_1_L19, but Isabelle's simplifier has problems using that result directly, so we reprove it specifically for integers.

lemma (in int0) Int_ZF_1_5_L7: assumes a $\in \mathbb{Z}_+$ and a$\leq$b
shows b $\in \mathbb{Z}_+$
⟨*proof*⟩

Adding a positive integer increases integers.

lemma (in int0) Int_ZF_1_5_L7A: assumes a$\in\mathbb{Z}$ b $\in \mathbb{Z}_+$
shows a $\leq$ a+b a $\neq$ a+b a+b $\in \mathbb{Z}$
⟨*proof*⟩

For any integer m the greater of m and 1 is a positive integer that is greater or equal than m. If we add 1 to it we get a positive integer that is strictly greater than m.

lemma (in int0) Int˙ZF˙1˙5˙L7B: assumes a$\in\mathbb{Z}$
shows
a $\leq$ GreaterOf(IntegerOrder,**1**,a)
GreaterOf(IntegerOrder,**1**,a) $\in \mathbb{Z}_+$
GreaterOf(IntegerOrder,**1**,a) + **1** $\in \mathbb{Z}_+$
a $\leq$ GreaterOf(IntegerOrder,**1**,a) + **1**
a $\neq$ GreaterOf(IntegerOrder,**1**,a) + **1**
$\langle$*proof*$\rangle$

The opposite of an element of $\mathbb{Z}_+$ cannot belong to $\mathbb{Z}_+$.

lemma (in int0) Int˙ZF˙1˙5˙L8: assumes a $\in \mathbb{Z}_+$
shows (-a) $\notin \mathbb{Z}_+$
$\langle$*proof*$\rangle$

For every integer there is one in $\mathbb{Z}_+$ that is greater or equal.

lemma (in int0) Int˙ZF˙1˙5˙L9: assumes a$\in\mathbb{Z}$
shows $\exists$b$\in\mathbb{Z}_+$. a$\leq$b
$\langle$*proof*$\rangle$

A theorem about odd extensions. Recall from OrdereGroup˙ZF.thy that the odd extension of an integer function f defined on $\mathbb{Z}_+$ is the odd function on $\mathbb{Z}$ equal to f on $\mathbb{Z}_+$. First we show that the odd extension is defined on $\mathbb{Z}$.

lemma (in int0) Int˙ZF˙1˙5˙L10: assumes f : $\mathbb{Z}_+\rightarrow\mathbb{Z}$
shows OddExtension($\mathbb{Z}$,IntegerAddition,IntegerOrder,f) : $\mathbb{Z}\rightarrow\mathbb{Z}$
$\langle$*proof*$\rangle$

On $\mathbb{Z}_+$, the odd extension of f is the same as f.

lemma (in int0) Int˙ZF˙1˙5˙L11: assumes f : $\mathbb{Z}_+\rightarrow\mathbb{Z}$ and a $\in \mathbb{Z}_+$ and
g = OddExtension($\mathbb{Z}$,IntegerAddition,IntegerOrder,f)
shows g(a) = f(a)
$\langle$*proof*$\rangle$

On -$\mathbb{Z}_+$, the value of the odd extension of f is the negative of $f(-a)$.

lemma (in int0) Int˙ZF˙1˙5˙L12:
assumes f : $\mathbb{Z}_+\rightarrow\mathbb{Z}$ and a $\in$ (-$\mathbb{Z}_+$) and
g = OddExtension($\mathbb{Z}$,IntegerAddition,IntegerOrder,f)
shows g(a) = -(f(-a))
$\langle$*proof*$\rangle$

Odd extensions are odd on $\mathbb{Z}$.

lemma (in int0) int˙oddext˙is˙odd:
assumes f : $\mathbb{Z}_+\rightarrow\mathbb{Z}$ and a$\in\mathbb{Z}$ and
g = OddExtension($\mathbb{Z}$,IntegerAddition,IntegerOrder,f)
shows g(-a) = -(g(a))
$\langle$*proof*$\rangle$

Alternative definition of an odd function.

lemma (in int0) Int˙ZF˙1˙5˙L13: assumes A1: f: $\mathbb{Z}\rightarrow\mathbb{Z}$ shows
($\forall$ a$\in\mathbb{Z}$. f(-a) = (-f(a))) $\longleftrightarrow$ ($\forall$ a$\in\mathbb{Z}$. (-(f(-a))) = f(a))
⟨*proof*⟩

Another way of expressing the fact that odd extensions are odd.

lemma (in int0) int˙oddext˙is˙odd˙alt:
assumes f : $\mathbb{Z}_+\rightarrow\mathbb{Z}$ and a$\in\mathbb{Z}$ and
g = OddExtension($\mathbb{Z}$,IntegerAddition,IntegerOrder,f)
shows (-g(-a)) = g(a)
⟨*proof*⟩

43.6 Functions with infinite limits

In this section we consider functions (integer sequences) that have infinite limits. An integer function has infinite positive limit if it is arbitrarily large for large enough arguments. Similarly, a function has infinite negative limit if it is arbitrarily small for small enough arguments. The material in this come mostly from the section in OrderedGroup˙ZF.thy with he same title. Here we rewrite the theorems from that section in the notation we use for integers and add some results specific for the ordered group of integers.

If an image of a set by a function with infinite positive limit is bounded above, then the set itself is bounded above.

lemma (in int0) Int˙ZF˙1˙6˙L1: assumes f: $\mathbb{Z}\rightarrow\mathbb{Z}$ and
$\forall$ a$\in\mathbb{Z}$.$\exists$ b$\in\mathbb{Z}_+$.$\forall$ x. b$\leq$x $\longrightarrow$ a $\leq$ f(x) and A $\subseteq$ $\mathbb{Z}$ and
IsBoundedAbove(f(A),IntegerOrder)
shows IsBoundedAbove(A,IntegerOrder)
⟨*proof*⟩

If an image of a set defined by separation by a function with infinite positive limit is bounded above, then the set itself is bounded above.

lemma (in int0) Int˙ZF˙1˙6˙L2: assumes A1: X$\neq$0 and A2: f: $\mathbb{Z}\rightarrow\mathbb{Z}$ and
A3: $\forall$ a$\in\mathbb{Z}$.$\exists$ b$\in\mathbb{Z}_+$.$\forall$ x. b$\leq$x $\longrightarrow$ a $\leq$ f(x) and
A4: $\forall$ x$\in$X. b(x) $\in$ $\mathbb{Z}$ $\wedge$ f(b(x)) $\leq$ U
shows $\exists$ u.$\forall$ x$\in$X. b(x) $\leq$ u
⟨*proof*⟩

If an image of a set defined by separation by a integer function with infinite negative limit is bounded below, then the set itself is bounded above. This is dual to Int˙ZF˙1˙6˙L2.

lemma (in int0) Int˙ZF˙1˙6˙L3: assumes A1: X$\neq$0 and A2: f: $\mathbb{Z}\rightarrow\mathbb{Z}$ and
A3: $\forall$ a$\in\mathbb{Z}$.$\exists$ b$\in\mathbb{Z}_+$.$\forall$ y. b$\leq$y $\longrightarrow$ f(-y) $\leq$ a and
A4: $\forall$ x$\in$X. b(x) $\in$ $\mathbb{Z}$ $\wedge$ L $\leq$ f(b(x))
shows $\exists$ l.$\forall$ x$\in$X. l $\leq$ b(x)
⟨*proof*⟩

The next lemma combines Int˙ZF˙1˙6˙L2 and Int˙ZF˙1˙6˙L3 to show that if the image of a set defined by separation by a function with infinite limits is bounded, then the set itself is bounded. The proof again uses directly a fact from OrderedGroup˙ZF.

lemma (in int0) Int˙ZF˙1˙6˙L4:
  assumes A1: X≠0 and A2: f: ℤ→ℤ and
  A3: ∀a∈ℤ.∃b∈ℤ₊.∀x. b≤x ⟶ a ≤ f(x) and
  A4: ∀a∈ℤ.∃b∈ℤ₊.∀y. b≤y ⟶ f(-y) ≤ a and
  A5: ∀x∈X. b(x) ∈ ℤ ∧ f(b(x)) ≤ U ∧ L ≤ f(b(x))
  shows ∃M.∀x∈X. abs(b(x)) ≤ M
⟨*proof*⟩

If a function is larger than some constant for arguments large enough, then the image of a set that is bounded below is bounded below. This is not true for ordered groups in general, but only for those for which bounded sets are finite. This does not require the function to have infinite limit, but such functions do have this property.

lemma (in int0) Int˙ZF˙1˙6˙L5:
  assumes A1: f: ℤ→ℤ and A2: N∈ℤ and
  A3: ∀m. N≤m ⟶ L ≤ f(m) and
  A4: IsBoundedBelow(A,IntegerOrder)
  shows IsBoundedBelow(f(A),IntegerOrder)
⟨*proof*⟩

A function that has an infinite limit can be made arbitrarily large on positive integers by adding a constant. This does not actually require the function to have infinite limit, just to be larger than a constant for arguments large enough.

lemma (in int0) Int˙ZF˙1˙6˙L6: assumes A1: N∈ℤ and
  A2: ∀m. N≤m ⟶ L ≤ f(m) and
  A3: f: ℤ→ℤ and A4: K∈ℤ
  shows ∃c∈ℤ. ∀n∈ℤ₊. K ≤ f(n)+c
⟨*proof*⟩

If a function has infinite limit, then we can add such constant such that minimum of those arguments for which the function (plus the constant) is larger than another given constant is greater than a third constant. It is not as complicated as it sounds.

lemma (in int0) Int˙ZF˙1˙6˙L7:
  assumes A1: f: ℤ→ℤ and A2: K∈ℤ N∈ℤ and
  A3: ∀a∈ℤ.∃b∈ℤ₊.∀x. b≤x ⟶ a ≤ f(x)
  shows ∃C∈ℤ. N ≤ Minimum(IntegerOrder,–n∈ℤ₊. K ≤ f(n)+C″)
⟨*proof*⟩

For any integer m the function $k \mapsto m \cdot k$ has an infinite limit (or negative of that). This is why we put some properties of these functions here, even though they properly belong to a (yet nonexistent) section on homomorphisms. The next lemma shows that the set $\{a \cdot x : x \in Z\}$ can finite only if $a = 0$.

lemma (in int0) Int˙ZF˙1˙6˙L8:
assumes A1: a∈ℤ and A2: −a·x. x∈ℤ″ ∈ Fin(ℤ)
shows a = **0**
⟨*proof*⟩

43.7 Miscelaneous

In this section we put some technical lemmas needed in various other places that are hard to classify.

Suppose we have an integer expression (a meta-function)F such that $F(p)|p|$ is bounded by a linear function of $|p|$, that is for some integers A, B we have $F(p)|p| \leq A|p| + B$. We show that F is then bounded. The proof is easy, we just divide both sides by $|p|$ and take the limit (just kidding).

lemma (in int0) Int˙ZF˙1˙7˙L1:
assumes A1: ∀q∈ℤ. F(q) ∈ ℤ and
A2: ∀q∈ℤ. F(q)·abs(q) ≤ A·abs(q) + B and
A3: A∈ℤ B∈ℤ
shows ∃L. ∀p∈ℤ. F(p) ≤ L
⟨*proof*⟩

A lemma about splitting (not really, there is some overlap) the ℤ×ℤ into six subsets (cases). The subsets are as follows: first and third qaudrant, and second and fourth quadrant farther split by the $b = -a$ line.

lemma (in int0) int˙plane˙split˙in6: assumes a∈ℤ b∈ℤ
shows
0≤a ∧ **0**≤b ∨ a≤**0** ∧ b≤**0** ∨
a≤**0** ∧ **0**≤b ∧ **0** ≤ a+b ∨ a≤**0** ∧ **0**≤b ∧ a+b ≤ **0** ∨
0≤a ∧ b≤**0** ∧ **0** ≤ a+b ∨ **0**≤a ∧ b≤**0** ∧ a+b ≤ **0**
⟨*proof*⟩

end

44 Division on integers

theory IntDiv˙ZF˙IML imports Int˙ZF˙1 ZF.IntDiv

begin

This theory translates some results form the Isabelle's IntDiv.thy theory to the notation used by IsarMathLib.

44.1 Quotient and reminder

For any integers m, n , $n > 0$ there are unique integers q, p such that $0 \leq p < n$ and $m = n \cdot q + p$. Number p in this decompsition is usually called m

mod n. Standard Isabelle denotes numbers q, p as m zdiv n and m zmod n, resp., and we will use the same notation.

The next lemma is sometimes called the "quotient-reminder theorem".

lemma (in int0) IntDiv_ZF_1_L1: assumes m∈ℤ n∈ℤ
  shows m = n·(m zdiv n) + (m zmod n)
  ⟨*proof*⟩

If n is greater than 0 then m zmod n is between 0 and $n - 1$.

lemma (in int0) IntDiv_ZF_1_L2:
  assumes A1: m∈ℤ and A2: **0**≤n n≠**0**
  shows
  0 ≤ m zmod n
  m zmod n ≤ n   m zmod n ≠ n
  m zmod n ≤ n-**1**
⟨*proof*⟩

$(m \cdot k)$ div $k = m$.

lemma (in int0) IntDiv_ZF_1_L3:
  assumes m∈ℤ k∈ℤ and k≠**0**
  shows
  (m·k) zdiv k = m
  (k·m) zdiv k = m
  ⟨*proof*⟩

The next lemma essentially translates zdiv_mono1 from standard Isabelle to our notation.

lemma (in int0) IntDiv_ZF_1_L4:
  assumes A1: m ≤ k and A2: **0**≤n n≠**0**
  shows m zdiv n ≤ k zdiv n
⟨*proof*⟩

A quotient-reminder theorem about integers greater than a given product.

lemma (in int0) IntDiv_ZF_1_L5:
  assumes A1: n ∈ $\mathbb{Z}_+$ and A2: n ≤ k and A3: k·n ≤ m
  shows
  m = n·(m zdiv n) + (m zmod n)
  m = (m zdiv n)·n + (m zmod n)
  (m zmod n) ∈ **0**..(n-**1**)
  k ≤ (m zdiv n)
  m zdiv n ∈ $\mathbb{Z}_+$
⟨*proof*⟩

end

45 Integers 2

theory Int_ZF_2 imports func_ZF_1 Int_ZF_1 IntDiv_ZF_IML Group_ZF_3

begin

In this theory file we consider the properties of integers that are needed for the real numbers construction in Real_ZF series.

45.1 Slopes

In this section we study basic properties of slopes - the integer almost homomorphisms. The general definition of an almost homomorphism f on a group G written in additive notation requires the set $\{f(m+n) - f(m) - f(n) : m, n \in G\}$ to be finite. In this section we establish a definition that is equivalent for integers: that for all integer m, n we have $|f(m+n)-f(m)-f(n)| \leq L$ for some L.

First we extend the standard notation for integers with notation related to slopes. We define slopes as almost homomorphisms on the additive group of integers. The set of slopes is denoted $\mathcal{S}$. We also define "positive" slopes as those that take infinite number of positive values on positive integers. We write δ(s,m,n) to denote the homomorphism difference of s at m, n (i.e the expression $s(m+n) - s(m) - s(n)$). We denote maxδ(s) the maximum absolute value of homomorphism difference of s as m, n range over integers. If s is a slope, then the set of homomorphism differences is finite and this maximum exists. In Group_ZF_3 we define the equivalence relation on almost homomorphisms using the notion of a quotient group relation and use "$\approx$" to denote it. As here this symbol seems to be hogged by the standard Isabelle, we will use "$\sim$" instead "$\approx$". We show in this section that $s \sim r$ iff for some L we have $|s(m) - r(m)| \leq L$ for all integer m. The "$+$" denotes the first operation on almost homomorphisms. For slopes this is addition of functions defined in the natural way. The "$\circ$" symbol denotes the second operation on almost homomorphisms (see Group_ZF_3 for definition), defined for the group of integers. In short "$\circ$" is the composition of slopes. The "$^{-1}$" symbol acts as an infix operator that assigns the value $\min\{n \in Z_+ : p \leq f(n)\}$ to a pair (of sets) f and p. In application f represents a function defined on Z_+ and p is a positive integer. We choose this notation because we use it to construct the right inverse in the ring of classes of slopes and show that this ring is in fact a field. To study the homomorphism difference of the function defined by $p \mapsto f^{-1}(p)$ we introduce the symbol ε defined as $\varepsilon(f, \langle m, n\rangle) = f^{-1}(m+n) - f^{-1}(m) - f^{-1}(n)$. Of course the intention is to use the fact that $\varepsilon(f, \langle m, n\rangle)$ is the homomorphism difference of the function g defined as $g(m) = f^{-1}(m)$. We also define $\gamma(s, m, n)$ as the expression $\delta(f, m, -n) + s(0) - \delta(f, n, -n)$. This is useful because of the identity $f(m-n) = \gamma(m,n) + f(m) - f(n)$ that allows to obtain bounds on the value of a slope at the difference of of two integers. For every integer m we introduce notation m^S defined by $m^E(n) = m \cdot n$. The mapping $q \mapsto q^S$ embeds integers into $\mathcal{S}$ preserving the order, (that is, maps positive integers into $\mathcal{S}_+$).

locale int1 = int0 +

fixes slopes ($\mathcal{S}$)
defines slopes˙def[simp]: $\mathcal{S} \equiv$ AlmostHoms($\mathbb{Z}$,IntegerAddition)

fixes posslopes ($\mathcal{S}_+$)
defines posslopes˙def[simp]: $\mathcal{S}_+ \equiv$ –s$\in\mathcal{S}$. s($\mathbb{Z}_+$) $\cap$ $\mathbb{Z}_+ \notin$ Fin($\mathbb{Z}$)″

fixes δ
defines δ˙def[simp]: δ(s,m,n) $\equiv$ s(m+n)-s(m)-s(n)

fixes maxhomdiff (maxδ)
defines maxhomdiff˙def[simp]:
maxδ(s) $\equiv$ Maximum(IntegerOrder,–abs(δ(s,m,n)). $\langle$ m,n$\rangle$ $\in$ $\mathbb{Z}\times\mathbb{Z}$″)

fixes AlEqRel
defines AlEqRel˙def[simp]:
AlEqRel $\equiv$ QuotientGroupRel($\mathcal{S}$,AlHomOp1($\mathbb{Z}$,IntegerAddition),FinRangeFunctions($\mathbb{Z}$,$\mathbb{Z}$))

fixes AlEq (infix $\sim$ 68)
defines AlEq˙def[simp]: s $\sim$ r $\equiv$ $\langle$ s,r$\rangle$ $\in$ AlEqRel

fixes slope˙add (infix + 70)
defines slope˙add˙def[simp]: s + r $\equiv$ AlHomOp1($\mathbb{Z}$,IntegerAddition)$\langle$ s,r$\rangle$

fixes slope˙comp (infix $\circ$ 70)
defines slope˙comp˙def[simp]: s $\circ$ r $\equiv$ AlHomOp2($\mathbb{Z}$,IntegerAddition)$\langle$ s,r$\rangle$

fixes neg (-˙ [90] 91)
defines neg˙def[simp]: -s $\equiv$ GroupInv($\mathbb{Z}$,IntegerAddition) O s

fixes slope˙inv (infix $^{-1}$ 71)
defines slope˙inv˙def[simp]:
f^{-1}(p) $\equiv$ Minimum(IntegerOrder,–n$\in\mathbb{Z}_+$. p $\leq$ f(n)″)
fixes ε
defines ε˙def[simp]:
ε(f,p) $\equiv$ f^{-1}(fst(p)+snd(p)) - f^{-1}(fst(p)) - f^{-1}(snd(p))

fixes γ
defines γ˙def[simp]:
γ(s,m,n) $\equiv$ δ(s,m,-n) - δ(s,n,-n) + s($\mathbf{0}$)

fixes intembed ($^{\cdot S}$)

defines intembed˙def[simp]: $m^S \equiv -\langle n, m\cdot n\rangle.\ n\in\mathbb{Z}$″

We can use theorems proven in the group1 context.

lemma (in int1) Int˙ZF˙2˙1˙L1: shows group1($\mathbb{Z}$,IntegerAddition)
⟨*proof*⟩

Type information related to the homomorphism difference expression.

lemma (in int1) Int˙ZF˙2˙1˙L2: assumes $f\in\mathcal{S}$ and $n\in\mathbb{Z}$ $m\in\mathbb{Z}$
shows
$m+n \in \mathbb{Z}$
$f(m+n) \in \mathbb{Z}$
$f(m) \in \mathbb{Z}$ $f(n) \in \mathbb{Z}$
$f(m) + f(n) \in \mathbb{Z}$
HomDiff($\mathbb{Z}$,IntegerAddition,f,⟨ m,n⟩) $\in \mathbb{Z}$
⟨*proof*⟩

Type information related to the homomorphism difference expression.

lemma (in int1) Int˙ZF˙2˙1˙L2A:
assumes $f:\mathbb{Z}\rightarrow\mathbb{Z}$ and $n\in\mathbb{Z}$ $m\in\mathbb{Z}$
shows
$m+n \in \mathbb{Z}$
$f(m+n) \in \mathbb{Z}$ $f(m) \in \mathbb{Z}$ $f(n) \in \mathbb{Z}$
$f(m) + f(n) \in \mathbb{Z}$
HomDiff($\mathbb{Z}$,IntegerAddition,f,⟨ m,n⟩) $\in \mathbb{Z}$
⟨*proof*⟩

Slopes map integers into integers.

lemma (in int1) Int˙ZF˙2˙1˙L2B:
assumes A1: $f\in\mathcal{S}$ and A2: $m\in\mathbb{Z}$
shows $f(m) \in \mathbb{Z}$
⟨*proof*⟩

The homomorphism difference in multiplicative notation is defined as the expression $s(m\cdot n)\cdot(s(m)\cdot s(n))^{-1}$. The next lemma shows that in the additive notation used for integers the homomorphism difference is $f(m+n)-f(m)-f(n)$ which we denote as δ(f,m,n).

lemma (in int1) Int˙ZF˙2˙1˙L3:
assumes $f:\mathbb{Z}\rightarrow\mathbb{Z}$ and $m\in\mathbb{Z}$ $n\in\mathbb{Z}$
shows HomDiff($\mathbb{Z}$,IntegerAddition,f,⟨ m,n⟩) = δ(f,m,n)
⟨*proof*⟩

The next formula restates the definition of the homomorphism difference to express the value an almost homomorphism on a sum.

lemma (in int1) Int˙ZF˙2˙1˙L3A:
assumes A1: $f\in\mathcal{S}$ and A2: $m\in\mathbb{Z}$ $n\in\mathbb{Z}$
shows
$f(m+n) = f(m)+(f(n)+\delta(f,m,n))$

⟨*proof*⟩

The homomorphism difference of any integer function is integer.

lemma (in int1) Int˙ZF˙2˙1˙L3B:
  assumes f:$\mathbb{Z}\rightarrow\mathbb{Z}$ and m$\in\mathbb{Z}$ n$\in\mathbb{Z}$
  shows δ(f,m,n) $\in \mathbb{Z}$
  ⟨*proof*⟩

The value of an integer function at a sum expressed in terms of δ.

lemma (in int1) Int˙ZF˙2˙1˙L3C: assumes A1: f:$\mathbb{Z}\rightarrow\mathbb{Z}$ and A2: m$\in\mathbb{Z}$ n$\in\mathbb{Z}$
  shows f(m+n) = δ(f,m,n) + f(n) + f(m)
⟨*proof*⟩

The next lemma presents two ways the set of homomorphism differences can be written.

lemma (in int1) Int˙ZF˙2˙1˙L4: assumes A1: f:$\mathbb{Z}\rightarrow\mathbb{Z}$
  shows –abs(HomDiff($\mathbb{Z}$,IntegerAddition,f,x)). x $\in \mathbb{Z}\times\mathbb{Z}''$ =
  –abs(δ(f,m,n)). ⟨ m,n⟩ $\in \mathbb{Z}\times\mathbb{Z}''$
⟨*proof*⟩

If f maps integers into integers and for all $m, n \in Z$ we have $|f(m+n) - f(m) - f(n)| \leq L$ for some L, then f is a slope.

lemma (in int1) Int˙ZF˙2˙1˙L5: assumes A1: f:$\mathbb{Z}\rightarrow\mathbb{Z}$
  and A2: $\forall$m$\in\mathbb{Z}$.$\forall$n$\in\mathbb{Z}$. abs(δ(f,m,n)) $\leq$ L
  shows f$\in\mathcal{S}$
⟨*proof*⟩

The absolute value of homomorphism difference of a slope s does not exceed maxδ(s).

lemma (in int1) Int˙ZF˙2˙1˙L7:
  assumes A1: s$\in\mathcal{S}$ and A2: n$\in\mathbb{Z}$ m$\in\mathbb{Z}$
  shows
  abs(δ(s,m,n)) $\leq$ maxδ(s)
  δ(s,m,n) $\in \mathbb{Z}$ maxδ(s) $\in \mathbb{Z}$
  (-maxδ(s)) $\leq$ δ(s,m,n)
⟨*proof*⟩

A useful estimate for the value of a slope at 0, plus some type information for slopes.

lemma (in int1) Int˙ZF˙2˙1˙L8: assumes A1: s$\in\mathcal{S}$
  shows
  abs(s($\mathbf{0}$)) $\leq$ maxδ(s)
  $\mathbf{0}$ $\leq$ maxδ(s)
  abs(s($\mathbf{0}$)) $\in \mathbb{Z}$ maxδ(s) $\in \mathbb{Z}$
  abs(s($\mathbf{0}$)) + maxδ(s) $\in \mathbb{Z}$
⟨*proof*⟩

Int Group_ZF_3.thy we show that finite range functions valued in an abelian group form a normal subgroup of almost homomorphisms. This allows to define the equivalence relation between almost homomorphisms as the relation resulting from dividing by that normal subgroup. Then we show in Group_ZF_3_4_L12 that if the difference of f and g has finite range (actually $f(n) \cdot g(n)^{-1}$ as we use multiplicative notation in Group_ZF_3.thy), then f and g are equivalent. The next lemma translates that fact into the notation used in int1 context.

lemma (in int1) Int_ZF_2_1_L9: assumes A1: s∈$\mathcal{S}$ r∈$\mathcal{S}$
and A2: ∀m∈ℤ. abs(s(m)-r(m)) ≤ L
shows s ∼ r
⟨*proof*⟩

A neccessary condition for two slopes to be almost equal. For slopes the definition postulates the set $\{f(m) - g(m) : m \in Z\}$ to be finite. This lemma shows that this implies that $|f(m) - g(m)|$ is bounded (by some integer) as m varies over integers. We also mention here that in this context s ∼ r implies that both s and r are slopes.

lemma (in int1) Int_ZF_2_1_L9A: assumes s ∼ r
shows
∃L∈ℤ. ∀m∈ℤ. abs(s(m)-r(m)) ≤ L
s∈$\mathcal{S}$ r∈$\mathcal{S}$
⟨*proof*⟩

Let's recall that the relation of almost equality is an equivalence relation on the set of slopes.

lemma (in int1) Int_ZF_2_1_L9B: shows
AlEqRel ⊆ $\mathcal{S}$×$\mathcal{S}$
equiv($\mathcal{S}$,AlEqRel)
⟨*proof*⟩

Another version of sufficient condition for two slopes to be almost equal: if the difference of two slopes is a finite range function, then they are almost equal.

lemma (in int1) Int_ZF_2_1_L9C: assumes s∈$\mathcal{S}$ r∈$\mathcal{S}$ and
s + (-r) ∈ FinRangeFunctions(ℤ,ℤ)
shows
s ∼ r
r ∼ s
⟨*proof*⟩

If two slopes are almost equal, then the difference has finite range. This is the inverse of Int_ZF_2_1_L9C.

lemma (in int1) Int_ZF_2_1_L9D: assumes A1: s ∼ r
shows s + (-r) ∈ FinRangeFunctions(ℤ,ℤ)
⟨*proof*⟩

What is the value of a composition of slopes?

lemma (in int1) Int_ZF_2_1_L10:
  assumes $s\in\mathcal{S}$ $r\in\mathcal{S}$ and $m\in\mathbb{Z}$
  shows $(s\circ r)(m) = s(r(m))$ $s(r(m)) \in \mathbb{Z}$
  ⟨*proof*⟩

Composition of slopes is a slope.

lemma (in int1) Int_ZF_2_1_L11:
  assumes $s\in\mathcal{S}$ $r\in\mathcal{S}$
  shows $s\circ r \in \mathcal{S}$
  ⟨*proof*⟩

Negative of a slope is a slope.

lemma (in int1) Int_ZF_2_1_L12: assumes $s\in\mathcal{S}$ shows $-s \in \mathcal{S}$
  ⟨*proof*⟩

What is the value of a negative of a slope?

lemma (in int1) Int_ZF_2_1_L12A:
  assumes $s\in\mathcal{S}$ and $m\in\mathbb{Z}$ shows $(-s)(m) = -(s(m))$
  ⟨*proof*⟩

What are the values of a sum of slopes?

lemma (in int1) Int_ZF_2_1_L12B: assumes $s\in\mathcal{S}$ $r\in\mathcal{S}$ and $m\in\mathbb{Z}$
  shows $(s+r)(m) = s(m) + r(m)$
  ⟨*proof*⟩

Sum of slopes is a slope.

lemma (in int1) Int_ZF_2_1_L12C: assumes $s\in\mathcal{S}$ $r\in\mathcal{S}$
  shows $s+r \in \mathcal{S}$
  ⟨*proof*⟩

A simple but useful identity.

lemma (in int1) Int_ZF_2_1_L13:
  assumes $s\in\mathcal{S}$ and $n\in\mathbb{Z}$ $m\in\mathbb{Z}$
  shows $s(n\cdot m) + (s(m) + \delta(s,n\cdot m,m)) = s((n+\mathbf{1})\cdot m)$
  ⟨*proof*⟩

Some estimates for the absolute value of a slope at the opposite integer.

lemma (in int1) Int_ZF_2_1_L14: assumes A1: $s\in\mathcal{S}$ and A2: $m\in\mathbb{Z}$
  shows
  $s(-m) = s(\mathbf{0}) - \delta(s,m,-m) - s(m)$
  $abs(s(m)+s(-m)) \leq \mathbf{2}\cdot max\delta(s)$
  $abs(s(-m)) \leq \mathbf{2}\cdot max\delta(s) + abs(s(m))$
  $s(-m) \leq abs(s(\mathbf{0})) + max\delta(s) - s(m)$
⟨*proof*⟩

An identity that expresses the value of an integer function at the opposite integer in terms of the value of that function at the integer, zero, and the homomorphism difference. We have a similar identity in Int_ZF_2_1_L14, but over there we assume that f is a slope.

lemma (in int1) Int_ZF_2_1_L14A: assumes A1: f:$\mathbb{Z}\rightarrow\mathbb{Z}$ and A2: m$\in\mathbb{Z}$
shows f(-m) = (-δ(f,m,-m)) + f($\mathbf{0}$) - f(m)
⟨*proof*⟩

The next lemma allows to use the expression maxf(f,$\mathbf{0}$..M-1). Recall that maxf(f,A) is the maximum of (function) f on (the set) A.

lemma (in int1) Int_ZF_2_1_L15:
assumes s$\in\mathcal{S}$ and M $\in \mathbb{Z}_+$
shows
maxf(s,$\mathbf{0}$..(M-$\mathbf{1}$)) $\in \mathbb{Z}$
$\forall$n $\in$ $\mathbf{0}$..(M-$\mathbf{1}$). s(n) $\leq$ maxf(s,$\mathbf{0}$..(M-$\mathbf{1}$))
minf(s,$\mathbf{0}$..(M-$\mathbf{1}$)) $\in \mathbb{Z}$
$\forall$n $\in$ $\mathbf{0}$..(M-$\mathbf{1}$). minf(s,$\mathbf{0}$..(M-$\mathbf{1}$)) $\leq$ s(n)
⟨*proof*⟩

A lower estimate for the value of a slope at $nM + k$.

lemma (in int1) Int_ZF_2_1_L16:
assumes A1: s$\in\mathcal{S}$ and A2: m$\in\mathbb{Z}$ and A3: M $\in \mathbb{Z}_+$ and A4: k $\in$ $\mathbf{0}$..(M-$\mathbf{1}$)
shows s(m·M) + (minf(s,$\mathbf{0}$..(M-$\mathbf{1}$))- maxδ(s)) $\leq$ s(m·M+k)
⟨*proof*⟩

Identity is a slope.

lemma (in int1) Int_ZF_2_1_L17: shows id($\mathbb{Z}$) $\in \mathcal{S}$
⟨*proof*⟩

Simple identities about (absolute value of) homomorphism differences.

lemma (in int1) Int_ZF_2_1_L18:
assumes A1: f:$\mathbb{Z}\rightarrow\mathbb{Z}$ and A2: m$\in\mathbb{Z}$ n$\in\mathbb{Z}$
shows
abs(f(n) + f(m) - f(m+n)) = abs(δ(f,m,n))
abs(f(m) + f(n) - f(m+n)) = abs(δ(f,m,n))
(-(f(m))) - f(n) + f(m+n) = δ(f,m,n)
(-(f(n))) - f(m) + f(m+n) = δ(f,m,n)
abs((-f(m+n)) + f(m) + f(n)) = abs(δ(f,m,n))
⟨*proof*⟩

Some identities about the homomorphism difference of odd functions.

lemma (in int1) Int_ZF_2_1_L19:
assumes A1: f:$\mathbb{Z}\rightarrow\mathbb{Z}$ and A2: $\forall$x$\in\mathbb{Z}$. (-f(-x)) = f(x)
and A3: m$\in\mathbb{Z}$ n$\in\mathbb{Z}$
shows
abs(δ(f,-m,m+n)) = abs(δ(f,m,n))
abs(δ(f,-n,m+n)) = abs(δ(f,m,n))
δ(f,n,-(m+n)) = δ(f,m,n)
δ(f,m,-(m+n)) = δ(f,m,n)

abs(δ(f,-m,-n)) = abs(δ(f,m,n))
⟨*proof*⟩

Recall that f is a slope iff $f(m+n)-f(m)-f(n)$ is bounded as m,n ranges over integers. The next lemma is the first step in showing that we only need to check this condition as m,n ranges over positive intergers. Namely we show that if the condition holds for positive integers, then it holds if one integer is positive and the second one is nonnegative.

lemma (in int1) Int˙ZF˙2˙1˙L20: assumes A1: f:$\mathbb{Z}\rightarrow\mathbb{Z}$ and
A2: $\forall$a$\in\mathbb{Z}_+$. $\forall$b$\in\mathbb{Z}_+$. abs(δ(f,a,b)) $\leq$ L and
A3: m$\in\mathbb{Z}^+$ n$\in\mathbb{Z}_+$
shows
0 $\leq$ L
abs(δ(f,m,n)) $\leq$ L + abs(f(**0**))
⟨*proof*⟩

If the slope condition holds for all pairs of integers such that one integer is positive and the second one is nonnegative, then it holds when both integers are nonnegative.

lemma (in int1) Int˙ZF˙2˙1˙L21: assumes A1: f:$\mathbb{Z}\rightarrow\mathbb{Z}$ and
A2: $\forall$a$\in\mathbb{Z}^+$. $\forall$b$\in\mathbb{Z}_+$. abs(δ(f,a,b)) $\leq$ L and
A3: n$\in\mathbb{Z}^+$ m$\in\mathbb{Z}^+$
shows abs(δ(f,m,n)) $\leq$ L + abs(f(**0**))
⟨*proof*⟩

If the homomorphism difference is bounded on $\mathbb{Z}_+\times\mathbb{Z}_+$, then it is bounded on $\mathbb{Z}^+\times\mathbb{Z}^+$.

lemma (in int1) Int˙ZF˙2˙1˙L22: assumes A1: f:$\mathbb{Z}\rightarrow\mathbb{Z}$ and
A2: $\forall$a$\in\mathbb{Z}_+$. $\forall$b$\in\mathbb{Z}_+$. abs(δ(f,a,b)) $\leq$ L
shows $\exists$M. $\forall$m$\in\mathbb{Z}^+$. $\forall$n$\in\mathbb{Z}^+$. abs(δ(f,m,n)) $\leq$ M
⟨*proof*⟩

For odd functions we can do better than in Int˙ZF˙2˙1˙L22: if the homomorphism difference of f is bounded on $\mathbb{Z}^+\times\mathbb{Z}^+$, then it is bounded on $\mathbb{Z}\times\mathbb{Z}$, hence f is a slope. Loong prof by splitting the $\mathbb{Z}\times\mathbb{Z}$ into six subsets.

lemma (in int1) Int˙ZF˙2˙1˙L23: assumes A1: f:$\mathbb{Z}\rightarrow\mathbb{Z}$ and
A2: $\forall$a$\in\mathbb{Z}_+$. $\forall$b$\in\mathbb{Z}_+$. abs(δ(f,a,b)) $\leq$ L
and A3: $\forall$x$\in\mathbb{Z}$. (-f(-x)) = f(x)
shows f$\in\mathcal{S}$
⟨*proof*⟩

If the homomorphism difference of a function defined on positive integers is bounded, then the odd extension of this function is a slope.

lemma (in int1) Int˙ZF˙2˙1˙L24:
assumes A1: f:$\mathbb{Z}_+\rightarrow\mathbb{Z}$ and A2: $\forall$a$\in\mathbb{Z}_+$. $\forall$b$\in\mathbb{Z}_+$. abs(δ(f,a,b)) $\leq$ L
shows OddExtension($\mathbb{Z}$,IntegerAddition,IntegerOrder,f) $\in\mathcal{S}$
⟨*proof*⟩

Type information related to γ.

lemma (in int1) Int˙ZF˙2˙1˙L25:
  assumes A1: f:ℤ→ℤ and A2: m∈ℤ n∈ℤ
  shows
  δ(f,m,-n) ∈ ℤ
  δ(f,n,-n) ∈ ℤ
  (-δ(f,n,-n)) ∈ ℤ
  f(**0**) ∈ ℤ
  γ(f,m,n) ∈ ℤ
⟨*proof*⟩

A couple of formulae involving $f(m-n)$ and $\gamma(f,m,n)$.

lemma (in int1) Int˙ZF˙2˙1˙L26:
  assumes A1: f:ℤ→ℤ and A2: m∈ℤ n∈ℤ
  shows
  f(m-n) = γ(f,m,n) + f(m) - f(n)
  f(m-n) = γ(f,m,n) + (f(m) - f(n))
  f(m-n) + (f(n) - γ(f,m,n)) = f(m)
⟨*proof*⟩

A formula expressing the difference between $f(m-n-k)$ and $f(m)-f(n)-f(k)$ in terms of γ.

lemma (in int1) Int˙ZF˙2˙1˙L26A:
  assumes A1: f:ℤ→ℤ and A2: m∈ℤ n∈ℤ k∈ℤ
  shows
  f(m-n-k) - (f(m)- f(n) - f(k)) = γ(f,m-n,k) + γ(f,m,n)
⟨*proof*⟩

If s is a slope, then $\gamma(s,m,n)$ is uniformly bounded.

lemma (in int1) Int˙ZF˙2˙1˙L27: assumes A1: s∈𝒮
  shows ∃L∈ℤ. ∀m∈ℤ.∀n∈ℤ. abs(γ(s,m,n)) ≤ L
⟨*proof*⟩

If s is a slope, then $s(m) \leq s(m-1)+M$, where L does not depend on m.

lemma (in int1) Int˙ZF˙2˙1˙L28: assumes A1: s∈𝒮
  shows ∃M∈ℤ. ∀m∈ℤ. s(m) ≤ s(m-**1**) + M
⟨*proof*⟩

If s is a slope, then the difference between $s(m-n-k)$ and $s(m)-s(n)-s(k)$ is uniformly bounded.

lemma (in int1) Int˙ZF˙2˙1˙L29: assumes A1: s∈𝒮
  shows
  ∃M∈ℤ. ∀m∈ℤ.∀n∈ℤ.∀k∈ℤ. abs(s(m-n-k) - (s(m)-s(n)-s(k))) ≤M
⟨*proof*⟩

If s is a slope, then we can find integers M,K such that $s(m-n-k) \leq s(m)-s(n)-s(k)+M$ and $s(m)-s(n)-s(k)+K \leq s(m-n-k)$, for all integer m,n,k.

lemma (in int1) Int˙ZF˙2˙1˙L30: assumes A1: s∈$\mathcal{S}$
shows
∃M∈$\mathbb{Z}$. ∀m∈$\mathbb{Z}$.∀n∈$\mathbb{Z}$.∀k∈$\mathbb{Z}$. s(m-n-k) ≤ s(m)-s(n)-s(k)+M
∃K∈$\mathbb{Z}$. ∀m∈$\mathbb{Z}$.∀n∈$\mathbb{Z}$.∀k∈$\mathbb{Z}$. s(m)-s(n)-s(k)+K ≤ s(m-n-k)
⟨*proof*⟩

By definition functions f, g are almost equal if $f - g^*$ is bounded. In the next lemma we show it is sufficient to check the boundedness on positive integers.

lemma (in int1) Int˙ZF˙2˙1˙L31: assumes A1: s∈$\mathcal{S}$ r∈$\mathcal{S}$
and A2: ∀m∈$\mathbb{Z}_+$. abs(s(m)-r(m)) ≤ L
shows s ∼ r
⟨*proof*⟩

A sufficient condition for an odd slope to be almost equal to identity: If for all positive integers the value of the slope at m is between m and m plus some constant independent of m, then the slope is almost identity.

lemma (in int1) Int˙ZF˙2˙1˙L32: assumes A1: s∈$\mathcal{S}$ M∈$\mathbb{Z}$
and A2: ∀m∈$\mathbb{Z}_+$. m ≤ s(m) ∧ s(m) ≤ m+M
shows s ∼ id($\mathbb{Z}$)
⟨*proof*⟩

A lemma about adding a constant to slopes. This is actually proven in Group˙ZF˙3˙5˙L1, in Group˙ZF˙3.thy here we just refer to that lemma to show it in notation used for integers. Unfortunately we have to use raw set notation in the proof.

lemma (in int1) Int˙ZF˙2˙1˙L33:
assumes A1: s∈$\mathcal{S}$ and A2: c∈$\mathbb{Z}$ and
A3: r = −⟨m,s(m)+c⟩. m∈$\mathbb{Z}$″
shows
∀m∈$\mathbb{Z}$. r(m) = s(m)+c
r∈$\mathcal{S}$
s ∼ r
⟨*proof*⟩

45.2 Composing slopes

Composition of slopes is not commutative. However, as we show in this section if f and g are slopes then the range of $f \circ g - g \circ f$ is bounded. This allows to show that the multiplication of real numbers is commutative.

Two useful estimates.

lemma (in int1) Int˙ZF˙2˙2˙L1:
assumes A1: f:$\mathbb{Z}$→$\mathbb{Z}$ and A2: p∈$\mathbb{Z}$ q∈$\mathbb{Z}$
shows
abs(f((p+**1**)·q)-(p+**1**)·f(q)) ≤ abs(δ(f,p·q,q))+abs(f(p·q)-p·f(q))
abs(f((p-**1**)·q)-(p-**1**)·f(q)) ≤ abs(δ(f,(p-**1**)·q,q))+abs(f(p·q)-p·f(q))

⟨*proof*⟩

If f is a slope, then $|f(p \cdot q) - p \cdot f(q)| \leq (|p| + 1) \cdot \max\delta(\mathrm{f})$. The proof is by induction on p and the next lemma is the induction step for the case when $0 \leq p$.

lemma (in int1) Int_ZF_2_2_L2:
  assumes A1: f∈$\mathcal{S}$ and A2: **0**≤p q∈$\mathbb{Z}$
  and A3: abs(f(p·q)-p·f(q)) ≤ (abs(p)+**1**)·maxδ(f)
  shows
  abs(f((p+**1**)·q)-(p+**1**)·f(q)) ≤ (abs(p+**1**)+ **1**)·maxδ(f)
⟨*proof*⟩

If f is a slope, then $|f(p \cdot q) - p \cdot f(q)| \leq (|p| + 1) \cdot \max\delta$. The proof is by induction on p and the next lemma is the induction step for the case when $p \leq 0$.

lemma (in int1) Int_ZF_2_2_L3:
  assumes A1: f∈$\mathcal{S}$ and A2: p≤**0** q∈$\mathbb{Z}$
  and A3: abs(f(p·q)-p·f(q)) ≤ (abs(p)+**1**)·maxδ(f)
  shows abs(f((p-**1**)·q)-(p-**1**)·f(q)) ≤ (abs(p-**1**)+ **1**)·maxδ(f)
⟨*proof*⟩

If f is a slope, then $|f(p \cdot q) - p \cdot f(q)| \leq (|p| + 1) \cdot \max\delta(f)$. Proof by cases on $0 \leq p$.

lemma (in int1) Int_ZF_2_2_L4:
  assumes A1: f∈$\mathcal{S}$ and A2: p∈$\mathbb{Z}$ q∈$\mathbb{Z}$
  shows abs(f(p·q)-p·f(q)) ≤ (abs(p)+**1**)·maxδ(f)
⟨*proof*⟩

The next elegant result is Lemma 7 in the Arthan's paper [2].

lemma (in int1) Arthan_Lem_7:
 assumes A1: f∈$\mathcal{S}$ and A2: p∈$\mathbb{Z}$ q∈$\mathbb{Z}$
  shows abs(q·f(p)-p·f(q)) ≤ (abs(p)+abs(q)+**2**)·maxδ(f)
⟨*proof*⟩

This is Lemma 8 in the Arthan's paper.

lemma (in int1) Arthan_Lem_8: assumes A1: f∈$\mathcal{S}$
  shows ∃A B. A∈$\mathbb{Z}$ ∧ B∈$\mathbb{Z}$ ∧ (∀p∈$\mathbb{Z}$. abs(f(p)) ≤ A·abs(p)+B)
⟨*proof*⟩

If f and g are slopes, then $f \circ g$ is equivalent (almost equal) to $g \circ f$. This is Theorem 9 in Arthan's paper [2].

theorem (in int1) Arthan_Th_9: assumes A1: f∈$\mathcal{S}$ g∈$\mathcal{S}$
  shows f∘g ∼ g∘f
⟨*proof*⟩

end

46 Integers 3

theory Int_ZF_3 imports Int_ZF_2

begin

This theory is a continuation of Int_ZF_2. We consider here the properties of slopes (almost homomorphisms on integers) that allow to define the order relation and multiplicative inverse on real numbers. We also prove theorems that allow to show completeness of the order relation of real numbers we define in Real_ZF.

46.1 Positive slopes

This section provides background material for defining the order relation on real numbers.

Positive slopes are functions (of course.)

lemma (in int1) Int_ZF_2_3_L1: assumes A1: $f\in\mathcal{S}_+$ shows $f:\mathbb{Z}\rightarrow\mathbb{Z}$
⟨*proof*⟩

A small technical lemma to simplify the proof of the next theorem.

lemma (in int1) Int_ZF_2_3_L1A:
assumes A1: $f\in\mathcal{S}_+$ and A2: $\exists n \in f(\mathbb{Z}_+) \cap \mathbb{Z}_+.\ a\leq n$
shows $\exists M\in\mathbb{Z}_+.\ a \leq f(M)$
⟨*proof*⟩

The next lemma is Lemma 3 in the Arthan's paper.

lemma (in int1) Arthan_Lem_3:
assumes A1: $f\in\mathcal{S}_+$ and A2: $D \in \mathbb{Z}_+$
shows $\exists M\in\mathbb{Z}_+.\ \forall m\in\mathbb{Z}_+.\ (m+\mathbf{1})\cdot D \leq f(m\cdot M)$
⟨*proof*⟩

A special case of Arthan_Lem_3 when $D = 1$.

corollary (in int1) Arthan_L_3_spec: assumes A1: $f \in \mathcal{S}_+$
shows $\exists M\in\mathbb{Z}_+.\forall n\in\mathbb{Z}_+.\ n+\mathbf{1} \leq f(n\cdot M)$
⟨*proof*⟩

We know from Group_ZF_3.thy that finite range functions are almost homomorphisms. Besides reminding that fact for slopes the next lemma shows that finite range functions do not belong to $\mathcal{S}_+$. This is important, because the projection of the set of finite range functions defines zero in the real number construction in Real_ZF_x.thy series, while the projection of $\mathcal{S}_+$ becomes the set of (strictly) positive reals. We don't want zero to be positive, do we? The next lemma is a part of Lemma 5 in the Arthan's paper [2].

lemma (in int1) Int_ZF_2_3_L1B:

assumes A1: f $\in$ FinRangeFunctions($\mathbb{Z}$,$\mathbb{Z}$)
shows f$\in\mathcal{S}$ f $\notin \mathcal{S}_+$
⟨*proof*⟩

We want to show that if f is a slope and neither f nor $-f$ are in $\mathcal{S}_+$, then f is bounded. The next lemma is the first step towards that goal and shows that if slope is not in $\mathcal{S}_+$ then $f(\mathbb{Z}_+)$ is bounded above.

lemma (in int1) Int˙ZF˙2˙3˙L2: assumes A1: f$\in\mathcal{S}$ and A2: f $\notin \mathcal{S}_+$
shows IsBoundedAbove(f($\mathbb{Z}_+$), IntegerOrder)
⟨*proof*⟩

If f is a slope and $-f \notin \mathcal{S}_+$, then $f(\mathbb{Z}_+)$ is bounded below.

lemma (in int1) Int˙ZF˙2˙3˙L3: assumes A1: f$\in\mathcal{S}$ and A2: -f $\notin \mathcal{S}_+$
shows IsBoundedBelow(f($\mathbb{Z}_+$), IntegerOrder)
⟨*proof*⟩

A slope that is bounded on $\mathbb{Z}_+$ is bounded everywhere.

lemma (in int1) Int˙ZF˙2˙3˙L4:
assumes A1: f$\in\mathcal{S}$ and A2: m$\in\mathbb{Z}$
and A3: $\forall$n$\in\mathbb{Z}_+$. abs(f(n)) $\leq$ L
shows abs(f(m)) $\leq$ **2**$\cdot$maxδ(f) + L
⟨*proof*⟩

A slope whose image of the set of positive integers is bounded is a finite range function.

lemma (in int1) Int˙ZF˙2˙3˙L4A:
assumes A1: f$\in\mathcal{S}$ and A2: IsBounded(f($\mathbb{Z}_+$), IntegerOrder)
shows f $\in$ FinRangeFunctions($\mathbb{Z}$,$\mathbb{Z}$)
⟨*proof*⟩

A slope whose image of the set of positive integers is bounded below is a finite range function or a positive slope.

lemma (in int1) Int˙ZF˙2˙3˙L4B:
assumes f$\in\mathcal{S}$ and IsBoundedBelow(f($\mathbb{Z}_+$), IntegerOrder)
shows f $\in$ FinRangeFunctions($\mathbb{Z}$,$\mathbb{Z}$) $\vee$ f$\in\mathcal{S}_+$
⟨*proof*⟩

If one slope is not greater then another on positive integers, then they are almost equal or the difference is a positive slope.

lemma (in int1) Int˙ZF˙2˙3˙L4C: assumes A1: f$\in\mathcal{S}$ g$\in\mathcal{S}$ and
A2: $\forall$n$\in\mathbb{Z}_+$. f(n) $\leq$ g(n)
shows f$\sim$g $\vee$ g + (-f) $\in \mathcal{S}_+$
⟨*proof*⟩

Positive slopes are arbitrarily large for large enough arguments.

lemma (in int1) Int˙ZF˙2˙3˙L5:
assumes A1: f$\in\mathcal{S}_+$ and A2: K$\in\mathbb{Z}$

shows $\exists N\in\mathbb{Z}_+.\ \forall m.\ N\leq m \longrightarrow K \leq f(m)$
⟨*proof*⟩

Positive slopes are arbitrarily small for small enough arguments. Kind of dual to Int˙ZF˙2˙3˙L5.

lemma (in int1) Int˙ZF˙2˙3˙L5A: assumes A1: $f\in\mathcal{S}_+$ and A2: $K\in\mathbb{Z}$
shows $\exists N\in\mathbb{Z}_+.\ \forall m.\ N\leq m \longrightarrow f(-m) \leq K$
⟨*proof*⟩

A special case of Int˙ZF˙2˙3˙L5 where $K = 1$.

corollary (in int1) Int˙ZF˙2˙3˙L6: assumes $f\in\mathcal{S}_+$
shows $\exists N\in\mathbb{Z}_+.\ \forall m.\ N\leq m \longrightarrow f(m) \in \mathbb{Z}_+$
⟨*proof*⟩

A special case of Int˙ZF˙2˙3˙L5 where $m = N$.

corollary (in int1) Int˙ZF˙2˙3˙L6A: assumes $f\in\mathcal{S}_+$ and $K\in\mathbb{Z}$
shows $\exists N\in\mathbb{Z}_+.\ K \leq f(N)$
⟨*proof*⟩

If values of a slope are not bounded above, then the slope is positive.

lemma (in int1) Int˙ZF˙2˙3˙L7: assumes A1: $f\in\mathcal{S}$
and A2: $\forall K\in\mathbb{Z}.\ \exists n\in\mathbb{Z}_+.\ K \leq f(n)$
shows $f \in \mathcal{S}_+$
⟨*proof*⟩

For unbounded slope f either $f \in\mathcal{S}_+$ of $-f \in\mathcal{S}_+$.

theorem (in int1) Int˙ZF˙2˙3˙L8:
assumes A1: $f\in\mathcal{S}$ and A2: $f \notin$ FinRangeFunctions($\mathbb{Z}$,$\mathbb{Z}$)
shows $(f \in \mathcal{S}_+)$ Xor $((-f) \in \mathcal{S}_+)$
⟨*proof*⟩

The sum of positive slopes is a positive slope.

theorem (in int1) sum˙of˙pos˙sls˙is˙pos˙sl:
assumes A1: $f \in \mathcal{S}_+$ $g \in \mathcal{S}_+$
shows $f+g \in \mathcal{S}_+$
⟨*proof*⟩

The composition of positive slopes is a positive slope.

theorem (in int1) comp˙of˙pos˙sls˙is˙pos˙sl:
assumes A1: $f \in \mathcal{S}_+$ $g \in \mathcal{S}_+$
shows $f\circ g \in \mathcal{S}_+$
⟨*proof*⟩

A slope equivalent to a positive one is positive.

lemma (in int1) Int˙ZF˙2˙3˙L9:
assumes A1: $f \in \mathcal{S}_+$ and A2: $\langle f,g\rangle \in$ AlEqRel shows $g \in \mathcal{S}_+$
⟨*proof*⟩

The set of positive slopes is saturated with respect to the relation of equivalence of slopes.

lemma (in int1) pos˙slopes˙saturated: shows IsSaturated(AlEqRel,$\mathcal{S}_+$)
⟨*proof*⟩

A technical lemma involving a projection of the set of positive slopes and a logical epression with exclusive or.

lemma (in int1) Int˙ZF˙2˙3˙L10:
  assumes A1: f∈$\mathcal{S}$ g∈$\mathcal{S}$
  and A2: R = –AlEqRel–s″. s∈$\mathcal{S}_+$″
  and A3: (f∈$\mathcal{S}_+$) Xor (g∈$\mathcal{S}_+$)
  shows (AlEqRel–f″ ∈ R) Xor (AlEqRel–g″ ∈ R)
⟨*proof*⟩

Identity function is a positive slope.

lemma (in int1) Int˙ZF˙2˙3˙L11: shows id(ℤ) ∈ $\mathcal{S}_+$
⟨*proof*⟩

The identity function is not almost equal to any bounded function.

lemma (in int1) Int˙ZF˙2˙3˙L12: assumes A1: f ∈ FinRangeFunctions(ℤ,ℤ)
  shows ¬(id(ℤ) ∼ f)
⟨*proof*⟩

46.2 Inverting slopes

Not every slope is a 1:1 function. However, we can still invert slopes in the sense that if f is a slope, then we can find a slope g such that $f \circ g$ is almost equal to the identity function. The goal of this this section is to establish this fact for positive slopes.

If f is a positive slope, then for every positive integer p the set $\{n \in Z_+ : p \leq f(n)\}$ is a nonempty subset of positive integers. Recall that $f^{-1}(p)$ is the notation for the smallest element of this set.

lemma (in int1) Int˙ZF˙2˙4˙L1:
  assumes A1: f ∈ $\mathcal{S}_+$ and A2: p∈ℤ$_+$ and A3: A = –n∈ℤ$_+$. p ≤ f(n)″
  shows
  A ⊆ ℤ$_+$
  A ≠ 0
  f^{-1}(p) ∈ A
  ∀m∈A. f^{-1}(p) ≤ m
⟨*proof*⟩

If f is a positive slope and p is a positive integer p, then $f^{-1}(p)$ (defined as the minimum of the set $\{n \in Z_+ : p \leq f(n)\}$) is a (well defined) positive integer.

lemma (in int1) Int˙ZF˙2˙4˙L2:

assumes f $\in \mathcal{S}_+$ and p$\in \mathbb{Z}_+$
shows
$f^{-1}(p) \in \mathbb{Z}_+$
$p \leq f(f^{-1}(p))$
⟨*proof*⟩

If f is a positive slope and p is a positive integer such that $n \leq f(p)$, then $f^{-1}(n) \leq p$.

lemma (in int1) Int˙ZF˙2˙4˙L3:
assumes f $\in \mathcal{S}_+$ and m$\in \mathbb{Z}_+$ p$\in \mathbb{Z}_+$ and m $\leq$ f(p)
shows $f^{-1}(m) \leq p$
⟨*proof*⟩

An upper bound $f(f^{-1}(m) - 1)$ for positive slopes.

lemma (in int1) Int˙ZF˙2˙4˙L4:
assumes A1: f $\in \mathcal{S}_+$ and A2: m$\in \mathbb{Z}_+$ and A3: $f^{-1}(m)$-**1** $\in \mathbb{Z}_+$
shows $f(f^{-1}(m)$-**1**$) \leq m$ $f(f^{-1}(m)$-**1**$) \neq m$
⟨*proof*⟩

The (candidate for) the inverse of a positive slope is nondecreasing.

lemma (in int1) Int˙ZF˙2˙4˙L5:
assumes A1: f $\in \mathcal{S}_+$ and A2: m$\in \mathbb{Z}_+$ and A3: m$\leq$n
shows $f^{-1}(m) \leq f^{-1}(n)$
⟨*proof*⟩

If $f^{-1}(m)$ is positive and n is a positive integer, then, then $f^{-1}(m+n) - 1$ is positive.

lemma (in int1) Int˙ZF˙2˙4˙L6:
assumes A1: f $\in \mathcal{S}_+$ and A2: m$\in \mathbb{Z}_+$ n$\in \mathbb{Z}_+$ and
A3: $f^{-1}(m)$-**1** $\in \mathbb{Z}_+$
shows $f^{-1}(m+n)$-**1** $\in \mathbb{Z}_+$
⟨*proof*⟩

If f is a slope, then $f(f^{-1}(m+n) - f^{-1}(m) - f^{-1}(n))$ is uniformly bounded above and below. Will it be the messiest IsarMathLib proof ever? Only time will tell.

lemma (in int1) Int˙ZF˙2˙4˙L7: assumes A1: f $\in \mathcal{S}_+$ and
A2: $\forall$m$\in \mathbb{Z}_+$. $f^{-1}(m)$-**1** $\in \mathbb{Z}_+$
shows
$\exists$U$\in \mathbb{Z}$. $\forall$m$\in \mathbb{Z}_+$. $\forall$n$\in \mathbb{Z}_+$. $f(f^{-1}(m+n)-f^{-1}(m)-f^{-1}(n)) \leq U$
$\exists$N$\in \mathbb{Z}$. $\forall$m$\in \mathbb{Z}_+$. $\forall$n$\in \mathbb{Z}_+$. $N \leq f(f^{-1}(m+n)-f^{-1}(m)-f^{-1}(n))$
⟨*proof*⟩

The expression $f^{-1}(m+n) - f^{-1}(m) - f^{-1}(n)$ is uniformly bounded for all pairs $\langle m, n\rangle \in \mathbb{Z}_+ \times \mathbb{Z}_+$. Recall that in the int1 context ε(f,x) is defined so that $\varepsilon(f, \langle m, n\rangle) = f^{-1}(m+n) - f^{-1}(m) - f^{-1}(n)$.

lemma (in int1) Int˙ZF˙2˙4˙L8: assumes A1: f $\in \mathcal{S}_+$ and

A2: $\forall$m$\in\mathbb{Z}_+$. f^{-1}(m)-**1** $\in \mathbb{Z}_+$
shows $\exists$M. $\forall$x$\in\mathbb{Z}_+\times\mathbb{Z}_+$. abs($\varepsilon$(f,x)) $\leq$ M
$\langle$*proof*$\rangle$

The (candidate for) inverse of a positive slope is a (well defined) function on $\mathbb{Z}_+$.

lemma (in int1) Int˙ZF˙2˙4˙L9:
assumes A1: f $\in \mathcal{S}_+$ and A2: g = –$\langle$p,f^{-1}(p)$\rangle$. p$\in\mathbb{Z}_+$″
shows
g : $\mathbb{Z}_+\rightarrow\mathbb{Z}_+$
g : $\mathbb{Z}_+\rightarrow\mathbb{Z}$
$\langle$*proof*$\rangle$

What are the values of the (candidate for) the inverse of a positive slope?

lemma (in int1) Int˙ZF˙2˙4˙L10:
assumes A1: f $\in \mathcal{S}_+$ and A2: g = –$\langle$p,f^{-1}(p)$\rangle$. p$\in\mathbb{Z}_+$″ and A3: p$\in\mathbb{Z}_+$
shows g(p) = f^{-1}(p)
$\langle$*proof*$\rangle$

The (candidate for) the inverse of a positive slope is a slope.

lemma (in int1) Int˙ZF˙2˙4˙L11: assumes A1: f $\in \mathcal{S}_+$ and
A2: $\forall$m$\in\mathbb{Z}_+$. f^{-1}(m)-**1** $\in \mathbb{Z}_+$ and
A3: g = –$\langle$p,f^{-1}(p)$\rangle$. p$\in\mathbb{Z}_+$″
shows OddExtension($\mathbb{Z}$,IntegerAddition,IntegerOrder,g) $\in \mathcal{S}$
$\langle$*proof*$\rangle$

Every positive slope that is at least 2 on positive integers almost has an inverse.

lemma (in int1) Int˙ZF˙2˙4˙L12: assumes A1: f $\in \mathcal{S}_+$ and
A2: $\forall$m$\in\mathbb{Z}_+$. f^{-1}(m)-**1** $\in \mathbb{Z}_+$
shows $\exists$h$\in\mathcal{S}$. f$\circ$h $\sim$ id($\mathbb{Z}$)
$\langle$*proof*$\rangle$

Int˙ZF˙2˙4˙L12 is almost what we need, except that it has an assumption that the values of the slope that we get the inverse for are not smaller than 2 on positive integers. The Arthan's proof of Theorem 11 has a mistake where he says "note that for all but finitely many $m, n \in N$ $p = g(m)$ and $q = g(n)$ are both positive". Of course there may be infinitely many pairs $\langle m, n\rangle$ such that p, q are not both positive. This is however easy to workaround: we just modify the slope by adding a constant so that the slope is large enough on positive integers and then look for the inverse.

theorem (in int1) pos˙slope˙has˙inv: assumes A1: f $\in \mathcal{S}_+$
shows $\exists$g$\in\mathcal{S}$. f$\sim$g $\wedge$ ($\exists$h$\in\mathcal{S}$. g$\circ$h $\sim$ id($\mathbb{Z}$))
$\langle$*proof*$\rangle$

46.3 Completeness

In this section we consider properties of slopes that are needed for the proof of completeness of real numbers constructred in Real˙ZF˙1.thy. In particular we consider properties of embedding of integers into the set of slopes by the mapping $m \mapsto m^S$, where m^S is defined by $m^S(n) = m \cdot n$.

If m is an integer, then m^S is a slope whose value is $m \cdot n$ for every integer.

lemma (in int1) Int˙ZF˙2˙5˙L1: assumes A1: m ∈ $\mathbb{Z}$
  shows
  $\forall$ n ∈ $\mathbb{Z}$. (m^S)(n) = m·n
  m^S ∈ $\mathcal{S}$
⟨*proof*⟩

For any slope f there is an integer m such that there is some slope g that is almost equal to m^S and dominates f in the sense that $f \leq g$ on positive integers (which implies that either g is almost equal to f or $g-f$ is a positive slope. This will be used in Real˙ZF˙1.thy to show that for any real number there is an integer that (whose real embedding) is greater or equal.

lemma (in int1) Int˙ZF˙2˙5˙L2: assumes A1: f ∈ $\mathcal{S}$
  shows $\exists$ m∈$\mathbb{Z}$. $\exists$ g∈$\mathcal{S}$. (m^S∼g ∧ (f∼g ∨ g+(-f) ∈ $\mathcal{S}_+$))
⟨*proof*⟩

The negative of an integer embeds in slopes as a negative of the orgiginal embedding.

lemma (in int1) Int˙ZF˙2˙5˙L3: assumes A1: m ∈ $\mathbb{Z}$
  shows (-m)^S = -(m^S)
⟨*proof*⟩

The sum of embeddings is the embeding of the sum.

lemma (in int1) Int˙ZF˙2˙5˙L3A: assumes A1: m∈$\mathbb{Z}$ k∈$\mathbb{Z}$
  shows (m^S) + (k^S) = ((m+k)^S)
⟨*proof*⟩

The composition of embeddings is the embeding of the product.

lemma (in int1) Int˙ZF˙2˙5˙L3B: assumes A1: m∈$\mathbb{Z}$ k∈$\mathbb{Z}$
  shows (m^S) ∘ (k^S) = ((m·k)^S)
⟨*proof*⟩

Embedding integers in slopes preserves order.

lemma (in int1) Int˙ZF˙2˙5˙L4: assumes A1: m≤n
  shows (m^S) ∼ (n^S) ∨ (n^S)+(-(m^S)) ∈ $\mathcal{S}_+$
⟨*proof*⟩

We aim at showing that $m \mapsto m^S$ is an injection modulo the relation of almost equality. To do that we first show that if m^S has finite range, then $m = 0$.

lemma (in int1) Int_ZF_2_5_L5:
  assumes m∈$\mathbb{Z}$ and $m^S \in$ FinRangeFunctions($\mathbb{Z}$,$\mathbb{Z}$)
  shows m=$\mathbf{0}$
  ⟨*proof*⟩

Embeddings of two integers are almost equal only if the integers are equal.

lemma (in int1) Int_ZF_2_5_L6:
  assumes A1: m∈$\mathbb{Z}$ k∈$\mathbb{Z}$ and A2: $(m^S) \sim (k^S)$
  shows m=k
⟨*proof*⟩

Embedding of 1 is the identity slope and embedding of zero is a finite range function.

lemma (in int1) Int_ZF_2_5_L7: shows
  $\mathbf{1}^S$ = id($\mathbb{Z}$)
  $\mathbf{0}^S \in$ FinRangeFunctions($\mathbb{Z}$,$\mathbb{Z}$)
⟨*proof*⟩

A somewhat technical condition for a embedding of an integer to be "less or equal" (in the sense apriopriate for slopes) than the composition of a slope and another integer (embedding).

lemma (in int1) Int_ZF_2_5_L8:
  assumes A1: f ∈ $\mathcal{S}$ and A2: N ∈ $\mathbb{Z}$ M ∈ $\mathbb{Z}$ and
  A3: ∀n∈$\mathbb{Z}_+$. M·n ≤ f(N·n)
  shows $M^S \sim f\circ(N^S) \vee (f\circ(N^S)) + (-(M^S)) \in \mathcal{S}_+$
⟨*proof*⟩

Another technical condition for the composition of a slope and an integer (embedding) to be "less or equal" (in the sense apriopriate for slopes) than embedding of another integer.

lemma (in int1) Int_ZF_2_5_L9:
  assumes A1: f ∈ $\mathcal{S}$ and A2: N ∈ $\mathbb{Z}$ M ∈ $\mathbb{Z}$ and
  A3: ∀n∈$\mathbb{Z}_+$. f(N·n) ≤ M·n
  shows $f\circ(N^S) \sim (M^S) \vee (M^S) + (-(f\circ(N^S))) \in \mathcal{S}_+$
⟨*proof*⟩

end

47 Construction real numbers - the generic part

theory Real_ZF imports Int_ZF_IML Ring_ZF_1

begin

The goal of the Real_ZF series of theory files is to provide a contruction of the set of real numbers. There are several ways to construct real numbers. Most common start from the rational numbers and use Dedekind cuts or

Cauchy sequences. Real_ZF_x.thy series formalizes an alternative approach that constructs real numbers directly from the group of integers. Our formalization is mostly based on [2]. Different variants of this contruction are also described in [1] and [3]. I recommend to read these papers, but for the impatient here is a short description: we take a set of maps $s : Z \to Z$ such that the set $\{s(m+n)-s(m)-s(n)\}_{n,m\in Z}$ is finite (Z means the integers here). We call these maps slopes. Slopes form a group with the natural addition $(s+r)(n) = s(n)+r(n)$. The maps such that the set $s(Z)$ is finite (finite range functions) form a subgroup of slopes. The additive group of real numbers is defined as the quotient group of slopes by the (sub)group of finite range functions. The multiplication is defined as the projection of the composition of slopes into the resulting quotient (coset) space.

47.1 The definition of real numbers

This section contains the construction of the ring of real numbers as classes of slopes - integer almost homomorphisms. The real definitions are in Group_ZF_2 theory, here we just specialize the definitions of almost homomorphisms, their equivalence and operations to the additive group of integers from the general case of abelian groups considered in Group_ZF_2.

The set of slopes is defined as the set of almost homomorphisms on the additive group of integers.

definition
  Slopes ≡ AlmostHoms(int,IntegerAddition)

The first operation on slopes (pointwise addition) is a special case of the first operation on almost homomorphisms.

definition
  SlopeOp1 ≡ AlHomOp1(int,IntegerAddition)

The second operation on slopes (composition) is a special case of the second operation on almost homomorphisms.

definition
  SlopeOp2 ≡ AlHomOp2(int,IntegerAddition)

Bounded integer maps are functions from integers to integers that have finite range. They play a role of zero in the set of real numbers we are constructing.

definition
  BoundedIntMaps ≡ FinRangeFunctions(int,int)

Bounded integer maps form a normal subgroup of slopes. The equivalence relation on slopes is the (group) quotient relation defined by this subgroup.

definition
  SlopeEquivalenceRel ≡ QuotientGroupRel(Slopes,SlopeOp1,BoundedIntMaps)

The set of real numbers is the set of equivalence classes of slopes.

definition
  RealNumbers ≡ Slopes//SlopeEquivalenceRel

The addition on real numbers is defined as the projection of pointwise addition of slopes on the quotient. This means that the additive group of real numbers is the quotient group: the group of slopes (with pointwise addition) defined by the normal subgroup of bounded integer maps.

definition
  RealAddition ≡ ProjFun2(Slopes,SlopeEquivalenceRel,SlopeOp1)

Multiplication is defined as the projection of composition of slopes on the quotient. The fact that it works is probably the most surprising part of the construction.

definition
  RealMultiplication ≡ ProjFun2(Slopes,SlopeEquivalenceRel,SlopeOp2)

We first show that we can use theorems proven in some proof contexts (locales). The locale group1 requires assumption that we deal with an abelian group. The next lemma allows to use all theorems proven in the context called group1.

lemma Real˙ZF˙1˙L1: shows group1(int,IntegerAddition)
  ⟨*proof*⟩

Real numbers form a ring. This is a special case of the theorem proven in Ring˙ZF˙1.thy, where we show the same in general for almost homomorphisms rather than slopes.

theorem Real˙ZF˙1˙T1: shows IsAring(RealNumbers,RealAddition,RealMultiplication)
⟨*proof*⟩

We can use theorems proven in group0 and group1 contexts applied to the group of real numbers.

lemma Real˙ZF˙1˙L2: shows
  group0(RealNumbers,RealAddition)
  RealAddition –is commutative on″ RealNumbers
  group1(RealNumbers,RealAddition)
⟨*proof*⟩

Let's define some notation.

locale real0 =

  fixes real (ℝ)
  defines real˙def [simp]: ℝ ≡ RealNumbers

  fixes ra (infixl + 69)
  defines ra˙def [simp]: a+ b ≡ RealAddition⟨a,b⟩

fixes rminus (- ˙ 72)
defines rminus˙def [simp]:-a ≡ GroupInv(ℝ,RealAddition)(a)

fixes rsub (infixl - 69)
defines rsub˙def [simp]: a-b ≡ a+(-b)

fixes rm (infixl · 70)
defines rm˙def [simp]: a·b ≡ RealMultiplication⟨a,b⟩

fixes rzero (**0**)
defines rzero˙def [simp]:
0 ≡ TheNeutralElement(RealNumbers,RealAddition)

fixes rone (**1**)
defines rone˙def [simp]:
1 ≡ TheNeutralElement(RealNumbers,RealMultiplication)

fixes rtwo (**2**)
defines rtwo˙def [simp]: **2** ≡ **1**+**1**

fixes non˙zero ($\mathbb{R}_0$)
defines non˙zero˙def[simp]: $\mathbb{R}_0$ ≡ ℝ-–**0**˝

fixes inv ($^{˙-1}$ [90] 91)
defines inv˙def[simp]:
a^{-1} ≡ GroupInv($\mathbb{R}_0$,restrict(RealMultiplication,$\mathbb{R}_0 \times \mathbb{R}_0$))(a)

In real0 context all theorems proven in the ring0, context are valid.

lemma (in real0) Real˙ZF˙1˙L3: shows
ring0(ℝ,RealAddition,RealMultiplication)
⟨*proof*⟩

Lets try out our notation to see that zero and one are real numbers.

lemma (in real0) Real˙ZF˙1˙L4: shows **0**∈ℝ **1**∈ℝ
⟨*proof*⟩

The lemma below lists some properties that require one real number to state.

lemma (in real0) Real˙ZF˙1˙L5: assumes A1: a∈ℝ
shows
(-a) ∈ ℝ
(-(-a)) = a
a+**0** = a
0+a = a
a·**1** = a
1·a = a
a-a = **0**
a-**0** = a
⟨*proof*⟩

The lemma below lists some properties that require two real numbers to state.

lemma (in real0) Real˙ZF˙1˙L6: assumes a∈ℝ b∈ℝ
  shows
  a+b ∈ ℝ
  a-b ∈ ℝ
  a·b ∈ ℝ
  a+b = b+a
  (-a)·b = -(a·b)
  a·(-b) = -(a·b)
  ⟨*proof*⟩

Multiplication of reals is associative.

lemma (in real0) Real˙ZF˙1˙L6A: assumes a∈ℝ b∈ℝ c∈ℝ
  shows a·(b·c) = (a·b)·c
  ⟨*proof*⟩

Addition is distributive with respect to multiplication.

lemma (in real0) Real˙ZF˙1˙L7: assumes a∈ℝ b∈ℝ c∈ℝ
  shows
  a·(b+c) = a·b + a·c
  (b+c)·a = b·a + c·a
  a·(b-c) = a·b - a·c
  (b-c)·a = b·a - c·a
  ⟨*proof*⟩

A simple rearrangement with four real numbers.

lemma (in real0) Real˙ZF˙1˙L7A:
  assumes a∈ℝ b∈ℝ c∈ℝ d∈ℝ
  shows a-b + (c-d) = a+c-b-d
  ⟨*proof*⟩

RealAddition is defined as the projection of the first operation on slopes (that is, slope addition) on the quotient (slopes divided by the "almost equal" relation. The next lemma plays with definitions to show that this is the same as the operation induced on the appriopriate quotient group. The names AH, Op1 and FR are used in group1 context to denote almost homomorphisms, the first operation on AH and finite range functions resp.

lemma Real˙ZF˙1˙L8: assumes
  AH = AlmostHoms(int,IntegerAddition) and
  Op1 = AlHomOp1(int,IntegerAddition) and
  FR = FinRangeFunctions(int,int)
  shows RealAddition = QuotientGroupOp(AH,Op1,FR)
  ⟨*proof*⟩

The symbol **0** in the real0 context is defined as the neutral element of real addition. The next lemma shows that this is the same as the neutral element of the appriopriate quotient group.

lemma (in real0) Real˙ZF˙1˙L9: assumes
  AH = AlmostHoms(int,IntegerAddition) and
  Op1 = AlHomOp1(int,IntegerAddition) and
  FR = FinRangeFunctions(int,int) and
  r = QuotientGroupRel(AH,Op1,FR)
  shows
  TheNeutralElement(AH//r,QuotientGroupOp(AH,Op1,FR)) = **0**
  SlopeEquivalenceRel = r
  ⟨*proof*⟩

Zero is the class of any finite range function.

lemma (in real0) Real˙ZF˙1˙L10:
  assumes A1: s ∈ Slopes
  shows SlopeEquivalenceRel–s″ = **0** ⟷ s ∈ BoundedIntMaps
⟨*proof*⟩

We will need a couple of results from Group˙ZF˙3.thy The first two that state that the definition of addition and multiplication of real numbers are consistent, that is the result does not depend on the choice of the slopes representing the numbers. The second one implies that what we call SlopeEquivalenceRel is actually an equivalence relation on the set of slopes. We also show that the neutral element of the multiplicative operation on reals (in short number 1) is the class of the identity function on integers.

lemma Real˙ZF˙1˙L11: shows
  Congruent2(SlopeEquivalenceRel,SlopeOp1)
  Congruent2(SlopeEquivalenceRel,SlopeOp2)
  SlopeEquivalenceRel ⊆ Slopes × Slopes
  equiv(Slopes, SlopeEquivalenceRel)
  SlopeEquivalenceRel–id(int)″ =
  TheNeutralElement(RealNumbers,RealMultiplication)
  BoundedIntMaps ⊆ Slopes
⟨*proof*⟩

A one-side implication of the equivalence from Real˙ZF˙1˙L10: the class of a bounded integer map is the real zero.

lemma (in real0) Real˙ZF˙1˙L11A: assumes s ∈ BoundedIntMaps
  shows SlopeEquivalenceRel–s″ = **0**
  ⟨*proof*⟩

The next lemma is rephrases the result from Group˙ZF˙3.thy that says that the negative (the group inverse with respect to real addition) of the class of a slope is the class of that slope composed with the integer additive group inverse. The result and proof is not very readable as we use mostly generic set theory notation with long names here. Real˙ZF˙1.thy contains the same statement written in a more readable notation: $[-s] = -[s]$.

lemma (in real0) Real˙ZF˙1˙L12: assumes A1: s ∈ Slopes and
  Dr: r = QuotientGroupRel(Slopes,SlopeOp1,BoundedIntMaps)

shows r–GroupInv(int,IntegerAddition) O s″ = -(r–s″)
⟨*proof*⟩

Two classes are equal iff the slopes that represent them are almost equal.

lemma Real˙ZF˙1˙L13: assumes s ∈ Slopes p ∈ Slopes
and r = SlopeEquivalenceRel
shows r–s″ = r–p″ ⟷ ⟨s,p⟩ ∈ r
⟨*proof*⟩

Identity function on integers is a slope. Thislemma concludes the easy part of the construction that follows from the fact that slope equivalence classes form a ring. It is easy to see that multiplication of classes of almost homomorphisms is not commutative in general. The remaining properties of real numbers, like commutativity of multiplication and the existence of multiplicative inverses have to be proven using properties of the group of integers, rather that in general setting of abelian groups.

lemma Real˙ZF˙1˙L14: shows id(int) ∈ Slopes
⟨*proof*⟩

end

48 Construction of real numbers

theory Real˙ZF˙1 imports Real˙ZF Int˙ZF˙3 OrderedField˙ZF

begin

In this theory file we continue the construction of real numbers started in Real˙ZF to a succesful conclusion. We put here those parts of the construction that can not be done in the general settings of abelian groups and require integers.

48.1 Definitions and notation

In this section we define notions and notation needed for the rest of the construction.

We define positive slopes as those that take an infinite number of posititive values on the positive integers (see Int˙ZF˙2 for properties of positive slopes).

definition
PositiveSlopes ≡ –s ∈ Slopes.
s(PositiveIntegers) ∩ PositiveIntegers ∉ Fin(int)″

The order on the set of real numbers is constructed by specifying the set of positive reals. This set is defined as the projection of the set of positive slopes.

definition
 PositiveReals ≡ –SlopeEquivalenceRel–s″. s ∈ PositiveSlopes″

The order relation on real numbers is constructed from the set of positive elements in a standard way (see section "Alternative definitions" in OrderedGroup˙ZF.)

definition
 OrderOnReals ≡ OrderFromPosSet(RealNumbers,RealAddition,PositiveReals)

The next locale extends the locale real0 to define notation specific to the construction of real numbers. The notation follows the one defined in Int˙ZF˙2.thy. If m is an integer, then the real number which is the class of the slope $n \mapsto m \cdot n$ is denoted m^R. For a real number a notation $\lfloor a \rfloor$ means the largest integer m such that the real version of it (that is, m^R) is not greater than a. For an integer m and a subset of reals S the expression $\Gamma(S,m)$ is defined as $\max\{\lfloor p^R \cdot x \rfloor : x \in S\}$. This is plays a role in the proof of completeness of real numbers. We also reuse some notation defined in the int0 context, like $\mathbb{Z}_+$ (the set of positive integers) and abs(m) (the absolute value of an integer, and some defined in the int1 context, like the addition (+) and composition (∘ of slopes.

locale real1 = real0 +

 fixes AlEq (infix ∼ 68)
 defines AlEq˙def[simp]: s ∼ r ≡ ⟨s,r⟩ ∈ SlopeEquivalenceRel

 fixes slope˙add (infix + 70)
 defines slope˙add˙def[simp]:
 s + r ≡ SlopeOp1⟨s,r⟩

 fixes slope˙comp (infix ∘ 71)
 defines slope˙comp˙def[simp]: s ∘ r ≡ SlopeOp2⟨s,r⟩

 fixes slopes ($\mathcal{S}$)
 defines slopes˙def[simp]: $\mathcal{S}$ ≡ AlmostHoms(int,IntegerAddition)

 fixes posslopes ($\mathcal{S}_+$)
 defines posslopes˙def[simp]: $\mathcal{S}_+$ ≡ PositiveSlopes

 fixes slope˙class ([˙])
 defines slope˙class˙def[simp]: [f] ≡ SlopeEquivalenceRel–f″

 fixes slope˙neg (-˙ [90] 91)
 defines slope˙neg˙def[simp]: -s ≡ GroupInv(int,IntegerAddition) O s

 fixes lesseqr (infix ≤ 60)
 defines lesseqr˙def[simp]: a ≤ b ≡ ⟨a,b⟩ ∈ OrderOnReals

fixes sless (infix ¡ 60)
defines sless˙def[simp]: a ¡ b $\equiv$ a$\leq$b $\wedge$ a$\neq$b

fixes positivereals ($\mathbb{R}_+$)
defines positivereals˙def[simp]: $\mathbb{R}_+ \equiv$ PositiveSet($\mathbb{R}$,RealAddition,OrderOnReals)

fixes intembed ($\cdot^R$ [90] 91)
defines intembed˙def[simp]:
$m^R \equiv$ [–⟨n,IntegerMultiplication⟨m,n⟩ ⟩. n $\in$ int″]

fixes floor (⌊ ˙ ⌋)
defines floor˙def[simp]:
⌊a⌋ $\equiv$ Maximum(IntegerOrder,–m $\in$ int. $m^R \leq$ a″)

fixes Γ
defines Γ˙def[simp]: Γ(S,p) $\equiv$ Maximum(IntegerOrder,–⌊p^R·x⌋. x$\in$S″)

fixes ia (infixl + 69)
defines ia˙def[simp]: a+b $\equiv$ IntegerAddition⟨ a,b⟩

fixes iminus (- ˙ 72)
defines iminus˙def[simp]: -a $\equiv$ GroupInv(int,IntegerAddition)(a)

fixes isub (infixl - 69)
defines isub˙def[simp]: a-b $\equiv$ a+ (- b)

fixes intpositives ($\mathbb{Z}_+$)
defines intpositives˙def[simp]:
$\mathbb{Z}_+ \equiv$ PositiveSet(int,IntegerAddition,IntegerOrder)

fixes zlesseq (infix $\leq$ 60)
defines lesseq˙def[simp]: m $\leq$ n $\equiv$ ⟨m,n⟩ $\in$ IntegerOrder

fixes imult (infixl · 70)
defines imult˙def[simp]: a·b $\equiv$ IntegerMultiplication⟨ a,b⟩

fixes izero ($\mathbf{0}_Z$)
defines izero˙def[simp]: $\mathbf{0}_Z \equiv$ TheNeutralElement(int,IntegerAddition)

fixes ione ($\mathbf{1}_Z$)
defines ione˙def[simp]: $\mathbf{1}_Z \equiv$ TheNeutralElement(int,IntegerMultiplication)

fixes itwo ($\mathbf{2}_Z$)
defines itwo˙def[simp]: $\mathbf{2}_Z \equiv \mathbf{1}_Z+\mathbf{1}_Z$

fixes abs
defines abs˙def[simp]:
abs(m) $\equiv$ AbsoluteValue(int,IntegerAddition,IntegerOrder)(m)

fixes δ
defines δ˙def[simp]: δ(s,m,n) ≡ s(m+n)-s(m)-s(n)

48.2 Multiplication of real numbers

Multiplication of real numbers is defined as a projection of composition of slopes onto the space of equivalence classes of slopes. Thus, the product of the real numbers given as classes of slopes s and r is defined as the class of $s \circ r$. The goal of this section is to show that multiplication defined this way is commutative.

Let's recall a theorem from Int˙ZF˙2.thy that states that if f, g are slopes, then $f \circ g$ is equivalent to $g \circ f$. Here we conclude from that that the classes of $f \circ g$ and $g \circ f$ are the same.

lemma (in real1) Real˙ZF˙1˙1˙L2: assumes A1: f ∈ $\mathcal{S}$ g ∈ $\mathcal{S}$
shows [f∘g] = [g∘f]
⟨*proof*⟩

Classes of slopes are real numbers.

lemma (in real1) Real˙ZF˙1˙1˙L3: assumes A1: f ∈ $\mathcal{S}$
shows [f] ∈ ℝ
⟨*proof*⟩

Each real number is a class of a slope.

lemma (in real1) Real˙ZF˙1˙1˙L3A: assumes A1: a∈ℝ
shows ∃f∈$\mathcal{S}$. a = [f]
⟨*proof*⟩

It is useful to have the definition of addition and multiplication in the real1 context notation.

lemma (in real1) Real˙ZF˙1˙1˙L4:
assumes A1: f ∈ $\mathcal{S}$ g ∈ $\mathcal{S}$
shows
[f] + [g] = [f+g]
[f] · [g] = [f∘g]
⟨*proof*⟩

The next lemma is essentially the same as Real˙ZF˙1˙L12, but written in the notation defined in the real1 context. It states that if f is a slope, then $-[f] = [-f]$.

lemma (in real1) Real˙ZF˙1˙1˙L4A: assumes f ∈ $\mathcal{S}$
shows [-f] = -[f]
⟨*proof*⟩

Subtracting real numbers correspods to adding the opposite slope.

lemma (in real1) Real˙ZF˙1˙1˙L4B: assumes A1: f ∈ $\mathcal{S}$ g ∈ $\mathcal{S}$

shows [f] - [g] = [f+(-g)]
⟨*proof*⟩

Multiplication of real numbers is commutative.

theorem (in real1) real˙mult˙commute: assumes A1: a∈ℝ b∈ℝ
shows a·b = b·a
⟨*proof*⟩

Multiplication is commutative on reals.

lemma real˙mult˙commutative: shows
RealMultiplication –is commutative on″ RealNumbers
⟨*proof*⟩

The neutral element of multiplication of reals (denoted as **1** in the real1 context) is the class of identity function on integers. This is really shown in Real˙ZF˙1˙L11, here we only rewrite it in the notation used in the real1 context.

lemma (in real1) real˙one˙cl˙identity: shows [id(int)] = **1**
⟨*proof*⟩

If f is bounded, then its class is the neutral element of additive operation on reals (denoted as **0** in the real1 context).

lemma (in real1) real˙zero˙cl˙bounded˙map:
assumes f ∈ BoundedIntMaps shows [f] = **0**
⟨*proof*⟩

Two real numbers are equal iff the slopes that represent them are almost equal. This is proven in Real˙ZF˙1˙L13, here we just rewrite it in the notation used in the real1 context.

lemma (in real1) Real˙ZF˙1˙1˙L5:
assumes f ∈ $\mathcal{S}$ g ∈ $\mathcal{S}$
shows [f] = [g] ⟷ f ∼ g
⟨*proof*⟩

If the pair of function belongs to the slope equivalence relation, then their classes are equal. This is convenient, because we don't need to assume that f, g are slopes (follows from the fact that $f \sim g$).

lemma (in real1) Real˙ZF˙1˙1˙L5A: assumes f ∼ g
shows [f] = [g]
⟨*proof*⟩

Identity function on integers is a slope. This is proven in Real˙ZF˙1˙L13, here we just rewrite it in the notation used in the real1 context.

lemma (in real1) id˙on˙int˙is˙slope: shows id(int) ∈ $\mathcal{S}$
⟨*proof*⟩

A result from Int˙ZF˙2.thy: the identity function on integers is not almost equal to any bounded function.

lemma (in real1) Real˙ZF˙1˙1˙L7:
  assumes A1: f ∈ BoundedIntMaps
  shows ¬(id(int) ∼ f)
  ⟨*proof*⟩

Zero is not one.

lemma (in real1) real˙zero˙not˙one: shows **1**≠**0**
⟨*proof*⟩

Negative of a real number is a real number. Property of groups.

lemma (in real1) Real˙ZF˙1˙1˙L8: assumes a∈ℝ shows (-a) ∈ ℝ
  ⟨*proof*⟩

An identity with three real numbers.

lemma (in real1) Real˙ZF˙1˙1˙L9: assumes a∈ℝ b∈ℝ c∈ℝ
  shows a·(b·c) = a·c·b
  ⟨*proof*⟩

48.3 The order on reals

In this section we show that the order relation defined by prescribing the set of positive reals as the projection of the set of positive slopes makes the ring of real numbers into an ordered ring. We also collect the facts about ordered groups and rings that we use in the construction.

Positive slopes are slopes and positive reals are real.

lemma Real˙ZF˙1˙2˙L1: shows
  PositiveSlopes ⊆ Slopes
  PositiveReals ⊆ RealNumbers
⟨*proof*⟩

Positive reals are the same as classes of a positive slopes.

lemma (in real1) Real˙ZF˙1˙2˙L2:
  shows a ∈ PositiveReals ⟷ (∃ f∈$\mathcal{S}_+$. a = [f])
⟨*proof*⟩

Let's recall from Int˙ZF˙2.thy that the sum and composition of positive slopes is a positive slope.

lemma (in real1) Real˙ZF˙1˙2˙L3:
  assumes f∈$\mathcal{S}_+$ g∈$\mathcal{S}_+$
  shows
  f+g ∈ $\mathcal{S}_+$
  f∘g ∈ $\mathcal{S}_+$
  ⟨*proof*⟩

Bounded integer maps are not positive slopes.

lemma (in real1) Real˙ZF˙1˙2˙L5:
  assumes f ∈ BoundedIntMaps
  shows f ∉ $\mathcal{S}_+$
  ⟨*proof*⟩

The set of positive reals is closed under addition and multiplication. Zero (the neutral element of addition) is not a positive number.

lemma (in real1) Real˙ZF˙1˙2˙L6: shows
  PositiveReals –is closed under˝ RealAddition
  PositiveReals –is closed under˝ RealMultiplication
  0 ∉ PositiveReals
⟨*proof*⟩

If a class of a slope f is not zero, then either f is a positive slope or $-f$ is a positive slope. The real proof is in Int˙ZF˙2.thy.

lemma (in real1) Real˙ZF˙1˙2˙L7:
  assumes A1: f ∈ $\mathcal{S}$ and A2: [f] ≠ **0**
  shows (f ∈ $\mathcal{S}_+$) Xor ((-f) ∈ $\mathcal{S}_+$)
  ⟨*proof*⟩

The next lemma rephrases Int˙ZF˙2˙3˙L10 in the notation used in real1 context.

lemma (in real1) Real˙ZF˙1˙2˙L8:
  assumes A1: f ∈ $\mathcal{S}$ g ∈ $\mathcal{S}$
  and A2: (f ∈ $\mathcal{S}_+$) Xor (g ∈ $\mathcal{S}_+$)
  shows ([f] ∈ PositiveReals) Xor ([g] ∈ PositiveReals)
  ⟨*proof*⟩

The trichotomy law for the (potential) order on reals: if $a \neq 0$, then either a is positive or $-a$ is potitive.

lemma (in real1) Real˙ZF˙1˙2˙L9:
  assumes A1: a∈ℝ and A2: a≠**0**
  shows (a ∈ PositiveReals) Xor ((-a) ∈ PositiveReals)
⟨*proof*⟩

Finally we are ready to prove that real numbers form an ordered ring with no zero divisors.

theorem reals˙are˙ord˙ring: shows
  IsAnOrdRing(RealNumbers,RealAddition,RealMultiplication,OrderOnReals)
  OrderOnReals –is total on˝ RealNumbers
  PositiveSet(RealNumbers,RealAddition,OrderOnReals) = PositiveReals
  HasNoZeroDivs(RealNumbers,RealAddition,RealMultiplication)
⟨*proof*⟩

All theorems proven in the ring1 (about ordered rings), group3 (about ordered groups) and group1 (about groups) contexts are valid as applied to ordered real numbers with addition and (real) order.

lemma Real˙ZF˙1˙2˙L10: shows
ring1(RealNumbers,RealAddition,RealMultiplication,OrderOnReals)
IsAnOrdGroup(RealNumbers,RealAddition,OrderOnReals)
group3(RealNumbers,RealAddition,OrderOnReals)
OrderOnReals –is total on" RealNumbers
⟨*proof*⟩

If $a = b$ or $b - a$ is positive, then a is less or equal b.

lemma (in real1) Real˙ZF˙1˙2˙L11: assumes A1: a∈ℝ b∈ℝ and
A3: a=b ∨ b-a ∈ PositiveReals
shows a≤b
⟨*proof*⟩

A sufficient condition for two classes to be in the real order.

lemma (in real1) Real˙ZF˙1˙2˙L12: assumes A1: f ∈ $\mathcal{S}$ g ∈ $\mathcal{S}$ and
A2: f∼g ∨ (g + (-f)) ∈ $\mathcal{S}_+$
shows [f] ≤ [g]
⟨*proof*⟩

Taking negative on both sides reverses the inequality, a case with an inverse on one side. Property of ordered groups.

lemma (in real1) Real˙ZF˙1˙2˙L13:
assumes A1: a∈ℝ and A2: (-a) ≤ b
shows (-b) ≤ a
⟨*proof*⟩

Real order is antisymmetric.

lemma (in real1) real˙ord˙antisym:
assumes A1: a≤b b≤a shows a=b
⟨*proof*⟩

Real order is transitive.

lemma (in real1) real˙ord˙transitive: assumes A1: a≤b b≤c
shows a≤c
⟨*proof*⟩

We can multiply both sides of an inequality by a nonnegative real number.

lemma (in real1) Real˙ZF˙1˙2˙L14:
assumes a≤b and **0**≤c
shows
a·c ≤ b·c
c·a ≤ c·b
⟨*proof*⟩

A special case of Real˙ZF˙1˙2˙L14: we can multiply an inequality by a real number.

lemma (in real1) Real˙ZF˙1˙2˙L14A:

assumes A1: a≤b and A2: c∈$\mathbb{R}_+$
shows c·a ≤ c·b
⟨*proof*⟩

In the real1 context notation $a \leq b$ implies that a and b are real numbers.

lemma (in real1) Real˙ZF˙1˙2˙L15: assumes a≤b shows a∈$\mathbb{R}$ b∈$\mathbb{R}$
⟨*proof*⟩

$a \leq b$ implies that $0 \leq b - a$.

lemma (in real1) Real˙ZF˙1˙2˙L16: assumes a≤b
shows **0** ≤ b-a
⟨*proof*⟩

A sum of nonnegative elements is nonnegative.

lemma (in real1) Real˙ZF˙1˙2˙L17: assumes **0**≤a **0**≤b
shows **0** ≤ a+b
⟨*proof*⟩

We can add sides of two inequalities

lemma (in real1) Real˙ZF˙1˙2˙L18: assumes a≤b c≤d
shows a+c ≤ b+d
⟨*proof*⟩

The order on real is reflexive.

lemma (in real1) real˙ord˙refl: assumes a∈$\mathbb{R}$ shows a≤a
⟨*proof*⟩

We can add a real number to both sides of an inequality.

lemma (in real1) add˙num˙to˙ineq: assumes a≤b and c∈$\mathbb{R}$
shows a+c ≤ b+c
⟨*proof*⟩

We can put a number on the other side of an inequality, changing its sign.

lemma (in real1) Real˙ZF˙1˙2˙L19:
assumes a∈$\mathbb{R}$ b∈$\mathbb{R}$ and c ≤ a+b
shows c-b ≤ a
⟨*proof*⟩

What happens when one real number is not greater or equal than another?

lemma (in real1) Real˙ZF˙1˙2˙L20: assumes a∈$\mathbb{R}$ b∈$\mathbb{R}$ and ¬(a≤b)
shows b ¡ a
⟨*proof*⟩

We can put a number on the other side of an inequality, changing its sign, version with a minus.

lemma (in real1) Real˙ZF˙1˙2˙L21:
assumes a∈$\mathbb{R}$ b∈$\mathbb{R}$ and c ≤ a-b

shows c+b ≤ a
⟨proof⟩

The order on reals is a relation on reals.

lemma (in real1) Real˙ZF˙1˙2˙L22: shows OrderOnReals ⊆ $\mathbb{R}\times\mathbb{R}$
⟨proof⟩

A set that is bounded above in the sense defined by order on reals is a subset of real numbers.

lemma (in real1) Real˙ZF˙1˙2˙L23:
assumes A1: IsBoundedAbove(A,OrderOnReals)
shows A ⊆ $\mathbb{R}$
⟨proof⟩

Properties of the maximum of three real numbers.

lemma (in real1) Real˙ZF˙1˙2˙L24:
assumes A1: a∈$\mathbb{R}$ b∈$\mathbb{R}$ c∈$\mathbb{R}$
shows
Maximum(OrderOnReals,–a,b,c″) ∈ –a,b,c″
Maximum(OrderOnReals,–a,b,c″) ∈ $\mathbb{R}$
a ≤ Maximum(OrderOnReals,–a,b,c″)
b ≤ Maximum(OrderOnReals,–a,b,c″)
c ≤ Maximum(OrderOnReals,–a,b,c″)
⟨proof⟩

A form of transitivity for the order on reals.

lemma (in real1) real˙strict˙ord˙transit:
assumes A1: a≤b and A2: b¡c
shows a¡c
⟨proof⟩

We can multiply a right hand side of an inequality between positive real numbers by a number that is greater than one.

lemma (in real1) Real˙ZF˙1˙2˙L25:
assumes b ∈ $\mathbb{R}_+$ and a≤b and **1**¡c
shows a¡b·c
⟨proof⟩

We can move a real number to the other side of a strict inequality, changing its sign.

lemma (in real1) Real˙ZF˙1˙2˙L26:
assumes a∈$\mathbb{R}$ b∈$\mathbb{R}$ and a-b ¡ c
shows a ¡ c+b
⟨proof⟩

Real order is translation invariant.

lemma (in real1) real˙ord˙transl˙inv:

assumes a≤b and c∈ℝ
shows c+a ≤ c+b
⟨*proof*⟩

It is convenient to have the transitivity of the order on integers in the notation specific to real1 context. This may be confusing for the presentation readers: even though ≤ and ≤ are printed in the same way, they are different symbols in the source. In the real1 context the former denotes inequality between integers, and the latter denotes inequality between real numbers (classes of slopes). The next lemma is about transitivity of the order relation on integers.

lemma (in real1) int˙order˙transitive:
assumes A1: a≤b b≤c
shows a≤c
⟨*proof*⟩

A property of nonempty subsets of real numbers that don't have a maximum: for any element we can find one that is (strictly) greater.

lemma (in real1) Real˙ZF˙1˙2˙L27:
assumes A⊆ℝ and ¬HasAmaximum(OrderOnReals,A) and x∈A
shows ∃y∈A. x¡y
⟨*proof*⟩

The next lemma shows what happens when one real number is not greater or equal than another.

lemma (in real1) Real˙ZF˙1˙2˙L28:
assumes a∈ℝ b∈ℝ and ¬(a≤b)
shows b¡a
⟨*proof*⟩

If a real number is less than another, then the second one can not be less or equal that the first.

lemma (in real1) Real˙ZF˙1˙2˙L29:
assumes a¡b shows ¬(b≤a)
⟨*proof*⟩

48.4 Inverting reals

In this section we tackle the issue of existence of (multiplicative) inverses of real numbers and show that real numbers form an ordered field. We also restate here some facts specific to ordered fields that we need for the construction. The actual proofs of most of these facts can be found in Field˙ZF.thy and OrderedField˙ZF.thy

We rewrite the theorem from Int˙ZF˙2.thy that shows that for every positive slope we can find one that is almost equal and has an inverse.

lemma (in real1) pos˙slopes˙have˙inv: assumes f $\in \mathcal{S}_+$
shows $\exists$ g$\in\mathcal{S}$. f$\sim$g $\wedge$ ($\exists$ h$\in\mathcal{S}$. g$\circ$h $\sim$ id(int))
⟨*proof*⟩

The set of real numbers we are constructing is an ordered field.

theorem (in real1) reals˙are˙ord˙field: shows
IsAnOrdField(RealNumbers,RealAddition,RealMultiplication,OrderOnReals)
⟨*proof*⟩

Reals form a field.

lemma reals˙are˙field:
shows IsAfield(RealNumbers,RealAddition,RealMultiplication)
⟨*proof*⟩

Theorem proven in field0 and field1 contexts are valid as applied to real numbers.

lemma field˙cntxts˙ok: shows
field0(RealNumbers,RealAddition,RealMultiplication)
field1(RealNumbers,RealAddition,RealMultiplication,OrderOnReals)
⟨*proof*⟩

If a is positive, then a^{-1} is also positive.

lemma (in real1) Real˙ZF˙1˙3˙L1: assumes a $\in \mathbb{R}_+$
shows $a^{-1} \in \mathbb{R}_+$ $a^{-1} \in \mathbb{R}$
⟨*proof*⟩

A technical fact about multiplying strict inequality by the inverse of one of the sides.

lemma (in real1) Real˙ZF˙1˙3˙L2:
assumes a $\in \mathbb{R}_+$ and a^{-1} ¡ b
shows **1** ¡ b·a
⟨*proof*⟩

If a is smaller than b, then $(b-a)^{-1}$ is positive.

lemma (in real1) Real˙ZF˙1˙3˙L3: assumes a¡b
shows $(b\text{-}a)^{-1} \in \mathbb{R}_+$
⟨*proof*⟩

We can put a positive factor on the other side of a strict inequality, changing it to its inverse.

lemma (in real1) Real˙ZF˙1˙3˙L4:
assumes A1: a$\in\mathbb{R}$ b$\in\mathbb{R}_+$ and A2: a·b ¡ c
shows a ¡ c·b^{-1}
⟨*proof*⟩

We can put a positive factor on the other side of a strict inequality, changing it to its inverse, version with the product initially on the right hand side.

lemma (in real1) Real˙ZF˙1˙3˙L4A:
assumes A1: $b \in \mathbb{R}$ $c \in \mathbb{R}_+$ and A2: a ¡ b·c
shows $a \cdot c^{-1}$ ¡ b
⟨*proof*⟩

We can put a positive factor on the other side of an inequality, changing it to its inverse, version with the product initially on the right hand side.

lemma (in real1) Real˙ZF˙1˙3˙L4B:
assumes A1: $b \in \mathbb{R}$ $c \in \mathbb{R}_+$ and A2: $a \leq b \cdot c$
shows $a \cdot c^{-1} \leq b$
⟨*proof*⟩

We can put a positive factor on the other side of an inequality, changing it to its inverse, version with the product initially on the left hand side.

lemma (in real1) Real˙ZF˙1˙3˙L4C:
assumes A1: $a \in \mathbb{R}$ $b \in \mathbb{R}_+$ and A2: $a \cdot b \leq c$
shows $a \leq c \cdot b^{-1}$
⟨*proof*⟩

A technical lemma about solving a strict inequality with three real numbers and inverse of a difference.

lemma (in real1) Real˙ZF˙1˙3˙L5:
assumes a¡b and $(b\text{-}a)^{-1}$ ¡ c
shows $\mathbf{1}$ + a·c ¡ b·c
⟨*proof*⟩

We can multiply an inequality by the inverse of a positive number.

lemma (in real1) Real˙ZF˙1˙3˙L6:
assumes $a \leq b$ and $c \in \mathbb{R}_+$ shows $a \cdot c^{-1} \leq b \cdot c^{-1}$
⟨*proof*⟩

We can multiply a strict inequality by a positive number or its inverse.

lemma (in real1) Real˙ZF˙1˙3˙L7:
assumes a¡b and $c \in \mathbb{R}_+$ shows
a·c ¡ b·c
c·a ¡ c·b
$a \cdot c^{-1}$ ¡ $b \cdot c^{-1}$
⟨*proof*⟩

An identity with three real numbers, inverse and cancelling.

lemma (in real1) Real˙ZF˙1˙3˙L8: assumes$a \in \mathbb{R}$ $b \in \mathbb{R}$ $b \neq \mathbf{0}$ $c \in \mathbb{R}$
shows $a \cdot b \cdot (c \cdot b^{-1}) = a \cdot c$
⟨*proof*⟩

48.5 Completeness

This goal of this section is to show that the order on real numbers is complete, that is every subset of reals that is bounded above has a smallest upper bound.

If m is an integer, then m^R is a real number. Recall that in real1 context m^R denotes the class of the slope $n \mapsto m \cdot n$.

lemma (in real1) real˙int˙is˙real: assumes m ∈ int
  shows $\mathrm{m}^R \in \mathbb{R}$
  ⟨*proof*⟩

The negative of the real embedding of an integer is the embedding of the negative of the integer.

lemma (in real1) Real˙ZF˙1˙4˙L1: assumes m ∈ int
  shows $(\text{-m})^R = \text{-}(\mathrm{m}^R)$
  ⟨*proof*⟩

The embedding of sum of integers is the sum of embeddings.

lemma (in real1) Real˙ZF˙1˙4˙L1A: assumes m ∈ int k ∈ int
  shows $\mathrm{m}^R + \mathrm{k}^R = ((\mathrm{m}+\mathrm{k})^R)$
  ⟨*proof*⟩

The embedding of a difference of integers is the difference of embeddings.

lemma (in real1) Real˙ZF˙1˙4˙L1B: assumes A1: m ∈ int k ∈ int
  shows m^R - $\mathrm{k}^R = (\text{m-k})^R$
⟨*proof*⟩

The embedding of the product of integers is the product of embeddings.

lemma (in real1) Real˙ZF˙1˙4˙L1C: assumes m ∈ int k ∈ int
  shows $\mathrm{m}^R \cdot \mathrm{k}^R = (\mathrm{m}\cdot\mathrm{k})^R$
  ⟨*proof*⟩

For any real numbers there is an integer whose real version is greater or equal.

lemma (in real1) Real˙ZF˙1˙4˙L2: assumes A1: a∈ℝ
  shows ∃m∈int. a ≤ m^R
⟨*proof*⟩

For any real numbers there is an integer whose real version (embedding) is less or equal.

lemma (in real1) Real˙ZF˙1˙4˙L3: assumes A1: a∈ℝ
  shows –m ∈ int. $\mathrm{m}^R \leq \mathrm{a}'' \neq 0$
⟨*proof*⟩

Embeddings of two integers are equal only if the integers are equal.

lemma (in real1) Real˙ZF˙1˙4˙L4:
  assumes A1: m ∈ int k ∈ int and A2: $\mathrm{m}^R = \mathrm{k}^R$
  shows m=k
⟨*proof*⟩

The embedding of integers preserves the order.

lemma (in real1) Real˙ZF˙1˙4˙L5: assumes A1: m≤k
  shows $m^R \leq k^R$
⟨*proof*⟩

The embedding of integers preserves the strict order.

lemma (in real1) Real˙ZF˙1˙4˙L5A: assumes A1: m≤k m≠k
  shows m^R ¡ k^R
⟨*proof*⟩

For any real number there is a positive integer whose real version is (strictly) greater. This is Lemma 14 i) in [2].

lemma (in real1) Arthan˙Lemma14i: assumes A1: a∈ℝ
  shows ∃n∈$\mathbb{Z}_+$. a ¡ n^R
⟨*proof*⟩

If one embedding is less or equal than another, then the integers are also less or equal.

lemma (in real1) Real˙ZF˙1˙4˙L6:
  assumes A1: k ∈ int m ∈ int and A2: $m^R \leq k^R$
  shows m≤k
⟨*proof*⟩

The floor function is well defined and has expected properties.

lemma (in real1) Real˙ZF˙1˙4˙L7: assumes A1: a∈ℝ
  shows
  IsBoundedAbove(–m ∈ int. $m^R \leq$ a”,IntegerOrder)
  –m ∈ int. $m^R \leq$ a” ≠ 0
  ⌊a⌋ ∈ int
  $\lfloor a\rfloor^R \leq$ a
⟨*proof*⟩

Every integer whose embedding is less or equal a real number a is less or equal than the floor of a.

lemma (in real1) Real˙ZF˙1˙4˙L8:
  assumes A1: m ∈ int and A2: $m^R \leq$ a
  shows m ≤ ⌊a⌋
⟨*proof*⟩

Integer zero and one embed as real zero and one.

lemma (in real1) int˙0˙1˙are˙real˙zero˙one:
  shows $\mathbf{0}_Z{}^R = \mathbf{0}$ $\mathbf{1}_Z{}^R = \mathbf{1}$
  ⟨*proof*⟩

Integer two embeds as the real two.

lemma (in real1) int˙two˙is˙real˙two: shows $\mathbf{2}_Z{}^R = \mathbf{2}$
⟨*proof*⟩

A positive integer embeds as a positive (hence nonnegative) real.

lemma (in real1) int˙pos˙is˙real˙pos: assumes A1: $p\in\mathbb{Z}_+$
  shows
  $p^R \in \mathbb{R}$
  $\mathbf{0} \leq p^R$
  $p^R \in \mathbb{R}_+$
⟨*proof*⟩

The ordered field of reals we are constructing is archimedean, i.e., if x, y are its elements with y positive, then there is a positive integer M such that x is smaller than $M^R y$. This is Lemma 14 ii) in [2].

lemma (in real1) Arthan˙Lemma14ii: assumes A1: $x\in\mathbb{R}$ $y \in \mathbb{R}_+$
  shows $\exists M\in\mathbb{Z}_+.$ x ¡ $M^R\cdot y$
⟨*proof*⟩

Taking the floor function preserves the order.

lemma (in real1) Real˙ZF˙1˙4˙L9: assumes A1: $a\leq b$
  shows $\lfloor a\rfloor \leq \lfloor b\rfloor$
⟨*proof*⟩

If S is bounded above and p is a positive intereger, then $\Gamma(S, p)$ is well defined.

lemma (in real1) Real˙ZF˙1˙4˙L10:
  assumes A1: IsBoundedAbove(S,OrderOnReals) $S\neq 0$ and A2: $p\in\mathbb{Z}_+$
  shows
  IsBoundedAbove(−$\lfloor p^R\cdot x\rfloor$. $x\in S$″,IntegerOrder)
  $\Gamma(S,p) \in$ −$\lfloor p^R\cdot x\rfloor$. $x\in S$″
  $\Gamma(S,p) \in$ int
⟨*proof*⟩

If p is a positive integer, then for all $s \in S$ the floor of $p \cdot x$ is not greater that $\Gamma(S, p)$.

lemma (in real1) Real˙ZF˙1˙4˙L11:
  assumes A1: IsBoundedAbove(S,OrderOnReals) and A2: $x\in S$ and A3: $p\in\mathbb{Z}_+$
  shows $\lfloor p^R\cdot x\rfloor \leq \Gamma(S,p)$
⟨*proof*⟩

The candidate for supremum is an integer mapping with values given by Γ.

lemma (in real1) Real˙ZF˙1˙4˙L12:
  assumes A1: IsBoundedAbove(S,OrderOnReals) $S\neq 0$ and
  A2: g = −$\langle p,\Gamma(S,p)\rangle$. $p\in\mathbb{Z}_+$″
  shows
  $g : \mathbb{Z}_+\rightarrow$int
  $\forall n\in\mathbb{Z}_+.\ g(n) = \Gamma(S,n)$
⟨*proof*⟩

Every integer is equal to the floor of its embedding.

lemma (in real1) Real˙ZF˙1˙4˙L14: assumes A1: m $\in$ int
shows $\lfloor m^R \rfloor = m$
⟨*proof*⟩

Floor of (real) zero is (integer) zero.

lemma (in real1) floor˙01˙is˙zero˙one: shows
$\lfloor \mathbf{0} \rfloor = \mathbf{0}_Z \quad \lfloor \mathbf{1} \rfloor = \mathbf{1}_Z$
⟨*proof*⟩

Floor of (real) two is (integer) two.

lemma (in real1) floor˙2˙is˙two: shows $\lfloor \mathbf{2} \rfloor = \mathbf{2}_Z$
⟨*proof*⟩

Floor of a product of embeddings of integers is equal to the product of integers.

lemma (in real1) Real˙ZF˙1˙4˙L14A: assumes A1: m $\in$ int k $\in$ int
shows $\lfloor m^R \cdot k^R \rfloor = m \cdot k$
⟨*proof*⟩

Floor of the sum of a number and the embedding of an integer is the floor of the number plus the integer.

lemma (in real1) Real˙ZF˙1˙4˙L15: assumes A1: x$\in\mathbb{R}$ and A2: p $\in$ int
shows $\lfloor x + p^R \rfloor = \lfloor x \rfloor + p$
⟨*proof*⟩

Floor of the difference of a number and the embedding of an integer is the floor of the number minus the integer.

lemma (in real1) Real˙ZF˙1˙4˙L16: assumes A1: x$\in\mathbb{R}$ and A2: p $\in$ int
shows $\lfloor x - p^R \rfloor = \lfloor x \rfloor - p$
⟨*proof*⟩

The floor of sum of embeddings is the sum of the integers.

lemma (in real1) Real˙ZF˙1˙4˙L17: assumes m $\in$ int n $\in$ int
shows $\lfloor (m^R) + n^R \rfloor = m + n$
⟨*proof*⟩

A lemma about adding one to floor.

lemma (in real1) Real˙ZF˙1˙4˙L17A: assumes A1: a$\in\mathbb{R}$
shows $\mathbf{1} + \lfloor a \rfloor^R = (\mathbf{1}_Z + \lfloor a \rfloor)^R$
⟨*proof*⟩

The difference between the a number and the embedding of its floor is (strictly) less than one.

lemma (in real1) Real˙ZF˙1˙4˙L17B: assumes A1: a$\in\mathbb{R}$
shows
a - $\lfloor a \rfloor^R$ ¡ $\mathbf{1}$
a ¡ $(\mathbf{1}_Z + \lfloor a \rfloor)^R$

$\langle proof \rangle$

The next lemma corresponds to Lemma 14 iii) in [2]. It says that we can find a rational number between any two different real numbers.

lemma (in real1) Arthan˙Lemma14iii: assumes A1: x¡y
shows $\exists$M$\in$int. $\exists$N$\in\mathbb{Z}_+$. x$\cdot$N^R ¡ M^R $\wedge$ M^R ¡ y$\cdot$N^R
$\langle proof \rangle$

Some estimates for the homomorphism difference of the floor function.

lemma (in real1) Real˙ZF˙1˙4˙L18: assumes A1: x$\in$ℝ y$\in$ℝ
shows
abs($\lfloor$x+y$\rfloor$ - $\lfloor$x$\rfloor$ - $\lfloor$y$\rfloor$) $\leq$ $\mathbf{2}_Z$
$\langle proof \rangle$

Suppose $S \neq \emptyset$ is bounded above and $\Gamma(S, m) = \lfloor m^R \cdot x \rfloor$ for some positive integer m and $x \in S$. Then if $y \in S, x \leq y$ we also have $\Gamma(S, m) = \lfloor m^R \cdot y \rfloor$.

lemma (in real1) Real˙ZF˙1˙4˙L20:
assumes A1: IsBoundedAbove(S,OrderOnReals) S$\neq$0 and
A2: n$\in\mathbb{Z}_+$ x$\in$S and
A3: Γ(S,n) = $\lfloor$n$^R\cdot$x$\rfloor$ and
A4: y$\in$S x$\leq$y
shows Γ(S,n) = $\lfloor$n$^R\cdot$y$\rfloor$
$\langle proof \rangle$

The homomorphism difference of $n \mapsto \Gamma(S, n)$ is bounded by 2 on positive integers.

lemma (in real1) Real˙ZF˙1˙4˙L21:
assumes A1: IsBoundedAbove(S,OrderOnReals) S$\neq$0 and
A2: m$\in\mathbb{Z}_+$ n$\in\mathbb{Z}_+$
shows abs(Γ(S,m+n) - Γ(S,m) - Γ(S,n)) $\leq$ $\mathbf{2}_Z$
$\langle proof \rangle$

The next lemma provides sufficient condition for an odd function to be an almost homomorphism. It says for odd functions we only need to check that the homomorphism difference (denoted δ in the real1 context) is bounded on positive integers. This is really proven in Int˙ZF˙2.thy, but we restate it here for convenience. Recall from Group˙ZF˙3.thy that OddExtension of a function defined on the set of positive elements (of an ordered group) is the only odd function that is equal to the given one when restricted to positive elements.

lemma (in real1) Real˙ZF˙1˙4˙L21A:
assumes A1: f:$\mathbb{Z}_+\rightarrow$int $\forall$a$\in\mathbb{Z}_+$. $\forall$b$\in\mathbb{Z}_+$. abs(δ(f,a,b)) $\leq$ L
shows OddExtension(int,IntegerAddition,IntegerOrder,f) $\in \mathcal{S}$
$\langle proof \rangle$

The candidate for (a representant of) the supremum of a nonempty bounded above set is a slope.

lemma (in real1) Real˙ZF˙1˙4˙L22:

assumes A1: IsBoundedAbove(S,OrderOnReals) S≠0 and
A2: g = –⟨p,Γ(S,p)⟩. p∈$\mathbb{Z}_+$″
shows OddExtension(int,IntegerAddition,IntegerOrder,g) ∈ $\mathcal{S}$
⟨*proof*⟩

A technical lemma used in the proof that all elements of S are less or equal than the candidate for supremum of S.

lemma (in real1) Real˙ZF˙1˙4˙L23:
assumes A1: f ∈ $\mathcal{S}$ and A2: N ∈ int M ∈ int and
A3: ∀n∈$\mathbb{Z}_+$. M·n ≤ f(N·n)
shows M^R ≤ [f]·(N^R)
⟨*proof*⟩

A technical lemma aimed used in the proof the candidate for supremum of S is less or equal than any upper bound for S.

lemma (in real1) Real˙ZF˙1˙4˙L23A:
assumes A1: f ∈ $\mathcal{S}$ and A2: N ∈ int M ∈ int and
A3: ∀n∈$\mathbb{Z}_+$. f(N·n) ≤ M·n
shows [f]·(N^R) ≤ M^R
⟨*proof*⟩

The essential condition to claim that the candidate for supremum of S is greater or equal than all elements of S.

lemma (in real1) Real˙ZF˙1˙4˙L24:
assumes A1: IsBoundedAbove(S,OrderOnReals) and
A2: x¡y y∈S and
A4: N ∈ $\mathbb{Z}_+$ M ∈ int and
A5: M^R ¡ y·N^R and A6: p ∈ $\mathbb{Z}_+$
shows p·M ≤ Γ(S,p·N)
⟨*proof*⟩

An obvious fact about odd extension of a function $p \mapsto \Gamma(s,p)$ that is used a couple of times in proofs.

lemma (in real1) Real˙ZF˙1˙4˙L24A:
assumes A1: IsBoundedAbove(S,OrderOnReals) S≠0 and A2: p ∈ $\mathbb{Z}_+$
and A3:
h = OddExtension(int,IntegerAddition,IntegerOrder,–⟨p,Γ(S,p)⟩. p∈$\mathbb{Z}_+$″)
shows h(p) = Γ(S,p)
⟨*proof*⟩

The candidate for the supremum of S is not smaller than any element of S.

lemma (in real1) Real˙ZF˙1˙4˙L25:
assumes A1: IsBoundedAbove(S,OrderOnReals) and
A2: ¬HasAmaximum(OrderOnReals,S) and
A3: x∈S and A4:
h = OddExtension(int,IntegerAddition,IntegerOrder,–⟨p,Γ(S,p)⟩. p∈$\mathbb{Z}_+$″)
shows x ≤ [h]
⟨*proof*⟩

The essential condition to claim that the candidate for supremum of S is less or equal than any upper bound of S.

lemma (in real1) Real˙ZF˙1˙4˙L26:
  assumes A1: IsBoundedAbove(S,OrderOnReals) and
  A2: x≤y x∈S and
  A4: N ∈ $\mathbb{Z}_+$ M ∈ int and
  A5: y·N^R ¡ M^R and A6: p ∈ $\mathbb{Z}_+$
  shows ⌊(N·p)^R·x⌋ ≤ M·p
⟨*proof*⟩

A piece of the proof of the fact that the candidate for the supremum of S is not greater than any upper bound of S, done separately for clarity (of mind).

lemma (in real1) Real˙ZF˙1˙4˙L27:
  assumes IsBoundedAbove(S,OrderOnReals) S≠0 and
  h = OddExtension(int,IntegerAddition,IntegerOrder,–⟨p,Γ(S,p)⟩. p∈$\mathbb{Z}_+$″)
  and p ∈ $\mathbb{Z}_+$
  shows ∃x∈S. h(p) = ⌊p^R·x⌋
  ⟨*proof*⟩

The candidate for the supremum of S is not greater than any upper bound of S.

lemma (in real1) Real˙ZF˙1˙4˙L28:
  assumes A1: IsBoundedAbove(S,OrderOnReals) S≠0
  and A2: ∀x∈S. x≤y and A3:
  h = OddExtension(int,IntegerAddition,IntegerOrder,–⟨p,Γ(S,p)⟩. p∈$\mathbb{Z}_+$″)
  shows [h] ≤ y
⟨*proof*⟩

Now we can prove that every nonempty subset of reals that is bounded above has a supremum. Proof by considering two cases: when the set has a maximum and when it does not.

lemma (in real1) real˙order˙complete:
  assumes A1: IsBoundedAbove(S,OrderOnReals) S≠0
  shows HasAminimum(OrderOnReals,⋂a∈S. OrderOnReals–a″)
⟨*proof*⟩

Finally, we are ready to formulate the main result: that the construction of real numbers from the additive group of integers results in a complete ordered field. This theorem completes the construction. It was fun.

theorem eudoxus˙reals˙are˙reals: shows
  IsAmodelOfReals(RealNumbers,RealAddition,RealMultiplication,OrderOnReals)
  ⟨*proof*⟩

end

49 Complex numbers

theory Complex˙ZF imports func˙ZF˙1 OrderedField˙ZF

begin

The goal of this theory is to define complex numbers and prove that the Metamath complex numbers axioms hold.

49.1 From complete ordered fields to complex numbers

This section consists mostly of definitions and a proof context for talking about complex numbers. Suppose we have a set R with binary operations A and M and a relation r such that the quadruple (R, A, M, r) forms a complete ordered field. The next definitions take (R, A, M, r) and construct the sets that represent the structure of complex numbers: the carrier ($\mathbb{C} = R \times R$), binary operations of addition and multiplication of complex numbers and the order relation on $\mathbb{R} = R \times 0$. The ImCxAdd, ReCxAdd, ImCxMul, ReCxMul are helper meta-functions representing the imaginary part of a sum of complex numbers, the real part of a sum of real numbers, the imaginary part of a product of complex numbers and the real part of a product of real numbers, respectively. The actual operations (subsets of $(R \times R) \times R$ are named CplxAdd and CplxMul.

When R is an ordered field, it comes with an order relation. This induces a natural strict order relation on $\{\langle x, 0\rangle : x \in R\} \subseteq R \times R$. We call the set $\{\langle x, 0\rangle : x \in R\}$ ComplexReals(R,A) and the strict order relation CplxROrder(R,A,r). The order on the real axis of complex numbers is defined as the relation induced on it by the canonical projection on the first coordinate and the order we have on the real numbers. OK, lets repeat this slower. We start with the order relation r on a (model of) real numbers R. We want to define an order relation on a subset of complex numbers, namely on $R \times \{0\}$. To do that we use the notion of a relation induced by a mapping. The mapping here is $f : R \times \{0\} \to R, f\langle x, 0\rangle = x$ which is defined under a name of SliceProjection in func˙ZF.thy. This defines a relation r_1 (called InducedRelation(f,r), see func˙ZF) on $R \times \{0\}$ such that $\langle\langle x, 0\rangle, \langle y, 0\rangle \in r_1$ iff $\langle x, y\rangle \in r$. This way we get what we call CplxROrder(R,A,r). However, this is not the end of the story, because Metamath uses strict inequalities in its axioms, rather than weak ones like IsarMathLib (mostly). So we need to take the strict version of this order relation. This is done in the syntax definition of $<_{\mathbb{R}}$ in the definition of complex0 context. Since Metamath proves a lot of theorems about the real numbers extended with $+\infty$ and $-\infty$, we define the notation for inequalites on the extended real line as well.

A helper expression representing the real part of the sum of two complex numbers.

definition
ReCxAdd(R,A,a,b) ≡ A⟨fst(a),fst(b)⟩

An expression representing the imaginary part of the sum of two complex numbers.

definition
ImCxAdd(R,A,a,b) ≡ A⟨snd(a),snd(b)⟩

The set (function) that is the binary operation that adds complex numbers.

definition
CplxAdd(R,A) ≡
–⟨p, ⟨ ReCxAdd(R,A,fst(p),snd(p)),ImCxAdd(R,A,fst(p),snd(p)) ⟩ ⟩.
p∈(R×R)×(R×R)″

The expression representing the imaginary part of the product of complex numbers.

definition
ImCxMul(R,A,M,a,b) ≡ A⟨M⟨fst(a),snd(b)⟩, M⟨snd(a),fst(b)⟩ ⟩

The expression representing the real part of the product of complex numbers.

definition
ReCxMul(R,A,M,a,b) ≡
A⟨M⟨fst(a),fst(b)⟩,GroupInv(R,A)(M⟨snd(a),snd(b)⟩)⟩

The function (set) that represents the binary operation of multiplication of complex numbers.

definition
CplxMul(R,A,M) ≡
– ⟨p, ⟨ReCxMul(R,A,M,fst(p),snd(p)),ImCxMul(R,A,M,fst(p),snd(p))⟩ ⟩.
p ∈ (R×R)×(R×R)″

The definition real numbers embedded in the complex plane.

definition
ComplexReals(R,A) ≡ R×–TheNeutralElement(R,A)″

Definition of order relation on the real line.

definition
CplxROrder(R,A,r) ≡
InducedRelation(SliceProjection(ComplexReals(R,A)),r)

The next locale defines proof context and notation that will be used for complex numbers.

locale complex0 =
fixes R and A and M and r
assumes R˙are˙reals: IsAmodelOfReals(R,A,M,r)

fixes complex (ℂ)

defines complex˙def[simp]: $\mathbb{C} \equiv R \times R$

fixes rone ($\mathbf{1}_R$)
defines rone˙def[simp]: $\mathbf{1}_R \equiv$ TheNeutralElement(R,M)

fixes rzero ($\mathbf{0}_R$)
defines rzero˙def[simp]: $\mathbf{0}_R \equiv$ TheNeutralElement(R,A)

fixes one ($\mathbf{1}$)
defines one˙def[simp]: $\mathbf{1} \equiv \langle \mathbf{1}_R, \mathbf{0}_R \rangle$

fixes zero ($\mathbf{0}$)
defines zero˙def[simp]: $\mathbf{0} \equiv \langle \mathbf{0}_R, \mathbf{0}_R \rangle$

fixes iunit (i)
defines iunit˙def[simp]: $i \equiv \langle \mathbf{0}_R, \mathbf{1}_R \rangle$

fixes creal (IR)
defines creal˙def[simp]: IR $\equiv -\langle r, \mathbf{0}_R \rangle.\ r \in R''$

fixes rmul (infixl · 71)
defines rmul˙def[simp]: $a \cdot b \equiv M\langle a,b \rangle$

fixes radd (infixl + 69)
defines radd˙def[simp]: $a + b \equiv A\langle a,b \rangle$

fixes rneg (- ˙ 70)
defines rneg˙def[simp]: - a $\equiv$ GroupInv(R,A)(a)

fixes ca (infixl + 69)
defines ca˙def[simp]: $a + b \equiv$ CplxAdd(R,A)$\langle a,b \rangle$

fixes cm (infixl · 71)
defines cm˙def[simp]: $a \cdot b \equiv$ CplxMul(R,A,M)$\langle a,b \rangle$

fixes cdiv (infixl / 70)
defines cdiv˙def[simp]: $a / b \equiv \bigcup - x \in \mathbb{C}.\ b \cdot x = a\ ''$

fixes sub (infixl - 69)
defines sub˙def[simp]: a - b $\equiv \bigcup - x \in \mathbb{C}.\ b + x = a\ ''$

fixes cneg (-˙ 95)
defines cneg˙def[simp]: - a $\equiv$ $\mathbf{0}$ - a

fixes lessr (infix $<_{\mathbb{R}}$ 68)
defines lessr˙def[simp]:
$a <_{\mathbb{R}} b \equiv \langle a,b \rangle \in$ StrictVersion(CplxROrder(R,A,r))

fixes cpnf ($+\infty$)

defines cpnf˙def[simp]: $+\infty \equiv \mathbb{C}$

fixes cmnf $(-\infty)$
defines cmnf˙def[simp]: $-\infty \equiv$ –$\mathbb{C}$˝

fixes cxr $(\mathbb{R}^*)$
defines cxr˙def[simp]: $\mathbb{R}^* \equiv \mathbb{R} \cup$ –$+\infty$,$-\infty$˝

fixes cxn $(\mathbb{N})$
defines cxn˙def[simp]:
$\mathbb{N} \equiv \bigcap$ –N $\in$ Pow($\mathbb{R}$). $\mathbf{1} \in$ N $\wedge$ ($\forall$ n. n$\in$N $\longrightarrow$ n+$\mathbf{1} \in$ N)˝

fixes cltrrset (¡)
defines cltrrset˙def[simp]:
¡ $\equiv$ StrictVersion(CplxROrder(R,A,r)) $\cap$ $\mathbb{R}\times\mathbb{R}$ $\cup$
–$\langle-\infty,+\infty\rangle$˝ $\cup$ ($\mathbb{R}\times$–$+\infty$˝) $\cup$ (–$-\infty$˝$\times\mathbb{R}$)

fixes cltrr (infix ¡ 68)
defines cltrr˙def[simp]: a ¡ b $\equiv$ $\langle$a,b$\rangle$ $\in$ ¡

fixes lsq (infix $\leq$ 68)
defines lsq˙def[simp]: a $\leq$ b $\equiv$ $\neg$ (b ¡ a)

fixes two ($\mathbf{2}$)
defines two˙def[simp]: $\mathbf{2} \equiv \mathbf{1} + \mathbf{1}$

fixes three ($\mathbf{3}$)
defines three˙def[simp]: $\mathbf{3} \equiv \mathbf{2}+\mathbf{1}$

fixes four ($\mathbf{4}$)
defines four˙def[simp]: $\mathbf{4} \equiv \mathbf{3}+\mathbf{1}$

fixes five ($\mathbf{5}$)
defines five˙def[simp]: $\mathbf{5} \equiv \mathbf{4}+\mathbf{1}$

fixes six ($\mathbf{6}$)
defines six˙def[simp]: $\mathbf{6} \equiv \mathbf{5}+\mathbf{1}$

fixes seven ($\mathbf{7}$)
defines seven˙def[simp]: $\mathbf{7} \equiv \mathbf{6}+\mathbf{1}$

fixes eight ($\mathbf{8}$)
defines eight˙def[simp]: $\mathbf{8} \equiv \mathbf{7}+\mathbf{1}$

fixes nine ($\mathbf{9}$)
defines nine˙def[simp]: $\mathbf{9} \equiv \mathbf{8}+\mathbf{1}$

49.2 Axioms of complex numbers

In this section we will prove that all Metamath's axioms of complex numbers hold in the complex0 context.

The next lemma lists some contexts that are valid in the complex0 context.

lemma (in complex0) valid_cntxts: shows
  field1(R,A,M,r)
  field0(R,A,M)
  ring1(R,A,M,r)
  group3(R,A,r)
  ring0(R,A,M)
  M {is commutative on} R
  group0(R,A)
⟨*proof*⟩

The next lemma shows the definition of real and imaginary part of complex sum and product in a more readable form using notation defined in complex0 locale.

lemma (in complex0) cplx_mul_add_defs: shows
  ReCxAdd(R,A,⟨a,b⟩,⟨c,d⟩) = a + c
  ImCxAdd(R,A,⟨a,b⟩,⟨c,d⟩) = b + d
  ImCxMul(R,A,M,⟨a,b⟩,⟨c,d⟩) = a·d + b·c
  ReCxMul(R,A,M,⟨a,b⟩,⟨c,d⟩) = a·c + (-b·d)
⟨*proof*⟩

Real and imaginary parts of sums and products of complex numbers are real.

lemma (in complex0) cplx_mul_add_types:
  assumes A1: $z_1 \in \mathbb{C}$ $z_2 \in \mathbb{C}$
  shows
  ReCxAdd(R,A,z_1,z_2) ∈ R
  ImCxAdd(R,A,z_1,z_2) ∈ R
  ImCxMul(R,A,M,z_1,z_2) ∈ R
  ReCxMul(R,A,M,z_1,z_2) ∈ R
⟨*proof*⟩

Complex reals are complex. Recall the definition of $\mathbb{R}$ in the complex0 locale.

lemma (in complex0) axresscn: shows $\mathbb{R} \subseteq \mathbb{C}$
  ⟨*proof*⟩

Complex 1 is not complex 0.

lemma (in complex0) ax1ne0: shows $\mathbf{1} \neq \mathbf{0}$
⟨*proof*⟩

Complex addition is a complex valued binary operation on complex numbers.

lemma (in complex0) axaddopr: shows CplxAdd(R,A): $\mathbb{C} \times \mathbb{C} \to \mathbb{C}$

⟨*proof*⟩

Complex multiplication is a complex valued binary operation on complex numbers.

lemma (in complex0) axmulopr: shows CplxMul(R,A,M): $\mathbb{C} \times \mathbb{C} \to \mathbb{C}$
⟨*proof*⟩

What are the values of omplex addition and multiplication in terms of their real and imaginary parts?

lemma (in complex0) cplx˙mul˙add˙vals:
assumes A1: $a\in R$ $b\in R$ $c\in R$ $d\in R$
shows
$\langle a,b\rangle + \langle c,d\rangle = \langle a + c, b + d\rangle$
$\langle a,b\rangle \cdot \langle c,d\rangle = \langle a\cdot c + (-b\cdot d), a\cdot d + b\cdot c\rangle$
⟨*proof*⟩

Complex multiplication is commutative.

lemma (in complex0) axmulcom: assumes A1: $a \in \mathbb{C}$ $b \in \mathbb{C}$
shows $a\cdot b = b\cdot a$
⟨*proof*⟩

A sum of complex numbers is complex.

lemma (in complex0) axaddcl: assumes $a \in \mathbb{C}$ $b \in \mathbb{C}$
shows $a+b \in \mathbb{C}$
⟨*proof*⟩

A product of complex numbers is complex.

lemma (in complex0) axmulcl: assumes $a \in \mathbb{C}$ $b \in \mathbb{C}$
shows $a\cdot b \in \mathbb{C}$
⟨*proof*⟩

Multiplication is distributive with respect to addition.

lemma (in complex0) axdistr:
assumes A1: $a \in \mathbb{C}$ $b \in \mathbb{C}$ $c \in \mathbb{C}$
shows $a\cdot(b + c) = a\cdot b + a\cdot c$
⟨*proof*⟩

Complex addition is commutative.

lemma (in complex0) axaddcom: assumes $a \in \mathbb{C}$ $b \in \mathbb{C}$
shows $a+b = b+a$
⟨*proof*⟩

Complex addition is associative.

lemma (in complex0) axaddass: assumes A1: $a \in \mathbb{C}$ $b \in \mathbb{C}$ $c \in \mathbb{C}$
shows $a + b + c = a + (b + c)$
⟨*proof*⟩

Complex multiplication is associative.

lemma (in complex0) axmulass: assumes A1: a $\in \mathbb{C}$ b $\in \mathbb{C}$ c $\in \mathbb{C}$
shows a $\cdot$ b $\cdot$ c = a $\cdot$ (b $\cdot$ c)
$\langle proof \rangle$

Complex 1 is real. This really means that the pair $\langle 1, 0\rangle$ is on the real axis.

lemma (in complex0) ax1re: shows $\mathbf{1} \in \mathbb{R}$
$\langle proof \rangle$

The imaginary unit is a "square root" of -1 (that is, $i^2 + 1 = 0$).

lemma (in complex0) axi2m1: shows i$\cdot$i + $\mathbf{1}$ = $\mathbf{0}$
$\langle proof \rangle$

0 is the neutral element of complex addition.

lemma (in complex0) ax0id: assumes a $\in \mathbb{C}$
shows a + $\mathbf{0}$ = a
$\langle proof \rangle$

The imaginary unit is a complex number.

lemma (in complex0) axicn: shows i $\in \mathbb{C}$
$\langle proof \rangle$

All complex numbers have additive inverses.

lemma (in complex0) axnegex: assumes A1: a $\in \mathbb{C}$
shows $\exists$x$\in\mathbb{C}$. a + x = $\mathbf{0}$
$\langle proof \rangle$

A non-zero complex number has a multiplicative inverse.

lemma (in complex0) axrecex: assumes A1: a $\in \mathbb{C}$ and A2: a$\neq\mathbf{0}$
shows $\exists$x$\in\mathbb{C}$. a$\cdot$x = $\mathbf{1}$
$\langle proof \rangle$

Complex 1 is a right neutral element for multiplication.

lemma (in complex0) ax1id: assumes A1: a $\in \mathbb{C}$
shows a$\cdot\mathbf{1}$ = a
$\langle proof \rangle$

A formula for sum of (complex) real numbers.

lemma (in complex0) sum˙of˙reals: assumes a$\in\mathbb{R}$ b$\in\mathbb{R}$
shows
a + b = $\langle$fst(a) + fst(b),$\mathbf{0}_R\rangle$
$\langle proof \rangle$

The sum of real numbers is real.

lemma (in complex0) axaddrcl: assumes A1: a$\in\mathbb{R}$ b$\in\mathbb{R}$
shows a + b $\in \mathbb{R}$

⟨proof⟩

The formula for the product of (complex) real numbers.

lemma (in complex0) prod˙of˙reals: assumes A1: a∈ℝ b∈ℝ
shows a · b = ⟨fst(a)·fst(b),$\mathbf{0}_R$⟩
⟨proof⟩

The product of (complex) real numbers is real.

lemma (in complex0) axmulrcl: assumes a∈ℝ b∈ℝ
shows a · b ∈ ℝ
⟨proof⟩

The existence of a real negative of a real number.

lemma (in complex0) axrnegex: assumes A1: a∈ℝ
shows ∃ x ∈ ℝ. a + x = **0**
⟨proof⟩

Each nonzero real number has a real inverse

lemma (in complex0) axrrecex:
assumes A1: a ∈ ℝ a ≠ **0**
shows ∃x∈ℝ. a · x = **1**
⟨proof⟩

Our ℝ symbol is the real axis on the complex plane.

lemma (in complex0) real˙means˙real˙axis: shows ℝ = ComplexReals(R,A)
⟨proof⟩

The CplxROrder thing is a relation on the complex reals.

lemma (in complex0) cplx˙ord˙on˙cplx˙reals:
shows CplxROrder(R,A,r) ⊆ ℝ×ℝ
⟨proof⟩

The strict version of the complex relation is a relation on complex reals.

lemma (in complex0) cplx˙strict˙ord˙on˙cplx˙reals:
shows StrictVersion(CplxROrder(R,A,r)) ⊆ ℝ×ℝ
⟨proof⟩

The CplxROrder thing is a relation on the complex reals. Here this is formulated as a statement that in complex0 context $a < b$ implies that a, b are complex reals

lemma (in complex0) strict˙cplx˙ord˙type: assumes a $<_{\mathbb{R}}$ b
shows a∈ℝ b∈ℝ
⟨proof⟩

A more readable version of the definition of the strict order relation on the real axis. Recall that in the complex0 context r denotes the (non-strict) order relation on the underlying model of real numbers.

lemma (in complex0) def˙of˙real˙axis˙order: shows
$\langle x,\mathbf{0}_R\rangle <_{\mathbb{R}} \langle y,\mathbf{0}_R\rangle \longleftrightarrow \langle x,y\rangle \in r \wedge x\neq y$
⟨*proof*⟩

The (non strict) order on complex reals is antisymmetric, transitive and total.

lemma (in complex0) cplx˙ord˙antsym˙trans˙tot: shows
antisym(CplxROrder(R,A,r))
trans(CplxROrder(R,A,r))
CplxROrder(R,A,r) –is total on" ℝ
⟨*proof*⟩

The trichotomy law for the strict order on the complex reals.

lemma (in complex0) cplx˙strict˙ord˙trich:
assumes a ∈ ℝ b ∈ ℝ
shows Exactly˙1˙of˙3˙holds($a<_{\mathbb{R}}b$, a=b, $b<_{\mathbb{R}}a$)
⟨*proof*⟩

The strict order on the complex reals is kind of antisymetric.

lemma (in complex0) pre˙axlttri: assumes A1: a ∈ ℝ   b ∈ ℝ
shows $a <_{\mathbb{R}} b \longleftrightarrow \neg(a=b \vee b <_{\mathbb{R}} a)$
⟨*proof*⟩

The strict order on complex reals is transitive.

lemma (in complex0) cplx˙strict˙ord˙trans:
shows trans(StrictVersion(CplxROrder(R,A,r)))
⟨*proof*⟩

The strict order on complex reals is transitive - the explicit version of cplx˙strict˙ord˙trans.

lemma (in complex0) pre˙axlttrn:
assumes A1: $a <_{\mathbb{R}} b$ $b <_{\mathbb{R}} c$
shows $a <_{\mathbb{R}} c$
⟨*proof*⟩

The strict order on complex reals is preserved by translations.

lemma (in complex0) pre˙axltadd:
assumes A1: $a <_{\mathbb{R}} b$ and A2: c ∈ ℝ
shows $c+a <_{\mathbb{R}} c+b$
⟨*proof*⟩

The set of positive complex reals is closed with respect to multiplication.

lemma (in complex0) pre˙axmulgt0: assumes A1: $\mathbf{0} <_{\mathbb{R}} a$ $\mathbf{0} <_{\mathbb{R}} b$
shows $\mathbf{0} <_{\mathbb{R}} a\cdot b$
⟨*proof*⟩

The order on complex reals is linear and complete.

lemma (in complex0) cmplx˙reals˙ord˙lin˙compl: shows
CplxROrder(R,A,r) –is complete″
IsLinOrder($\mathbb{R}$,CplxROrder(R,A,r))
⟨*proof*⟩

The property of the strict order on complex reals that corresponds to completeness.

lemma (in complex0) pre˙axsup: assumes A1: X ⊆ $\mathbb{R}$ X ≠ 0 and
A2: ∃x∈$\mathbb{R}$. ∀y∈X. y $<_{\mathbb{R}}$ x
shows
∃x∈$\mathbb{R}$. (∀y∈X. ¬(x $<_{\mathbb{R}}$ y)) ∧ (∀y∈$\mathbb{R}$. (y $<_{\mathbb{R}}$ x ⟶ (∃z∈X. y $<_{\mathbb{R}}$ z)))
⟨*proof*⟩

end

50 Topology - introduction

theory Topology˙ZF imports ZF1 Finite˙ZF Fol1

begin

This theory file provides basic definitions and properties of topology, open and closed sets, closure and boundary.

50.1 Basic definitions and properties

A typical textbook defines a topology on a set X as a collection T of subsets of X such that $X \in T$, $\emptyset \in T$ and T is closed with respect to arbitrary unions and intersection of two sets. One can notice here that since we always have $\bigcup T = X$, the set on which the topology is defined (the "carrier" of the topology) can always be constructed from the topology itself and is superfluous in the definition. Moreover, as Marnix Klooster pointed out to me, the fact that the empty set is open can also be proven from other axioms. Hence, we define a topology as a collection of sets that is closed under arbitrary unions and intersections of two sets, without any mention of the set on which the topology is defined. Recall that Pow(T) is the powerset of T, so that if $M \in$ Pow(T) then M is a subset of T. The sets that belong to a topology T will be sometimes called "open in" T or just "open" if the topology is clear from the context.

Topology is a collection of sets that is closed under arbitrary unions and intersections of two sets.

definition
IsATopology (˙ –is a topology″ [90] 91) where
T –is a topology″ ≡ (∀M ∈ Pow(T). ⋃M ∈ T) ∧
(∀U∈T. ∀ V∈T. U∩V ∈ T)

We define interior of a set A as the union of all open sets contained in A. We use Interior(A,T) to denote the interior of A.

definition
Interior(A,T) ≡ $\bigcup$ –U∈T. U ⊆ A″

A set is closed if it is contained in the carrier of topology and its complement is open.

definition
IsClosed (infixl –is closed in″ 90) where
D –is closed in″ T ≡ (D ⊆ $\bigcup$T ∧ $\bigcup$T - D ∈ T)

To prove various properties of closure we will often use the collection of closed sets that contain a given set A. Such collection does not have a separate name in informal math. We will call it ClosedCovers(A,T).

definition
ClosedCovers(A,T) ≡ –D ∈ Pow($\bigcup$T). D –is closed in″ T ∧ A⊆D″

The closure of a set A is defined as the intersection of the collection of closed sets that contain A.

definition
Closure(A,T) ≡ $\bigcap$ ClosedCovers(A,T)

We also define boundary of a set as the intersection of its closure with the closure of the complement (with respect to the carrier).

definition
Boundary(A,T) ≡ Closure(A,T) ∩ Closure($\bigcup$T - A,T)

A set K is compact if for every collection of open sets that covers K we can choose a finite one that still covers the set. Recall that FinPow(M) is the collection of finite subsets of M (finite powerset of M), defined in IsarMathLib's Finite˙ZF theory.

definition
IsCompact (infixl –is compact in″ 90) where
K –is compact in″ T ≡ (K ⊆ $\bigcup$T ∧
(∀ M∈Pow(T). K ⊆ $\bigcup$M ⟶ (∃ N ∈ FinPow(M). K ⊆ $\bigcup$N)))

A basic example of a topology: the powerset of any set is a topology.

lemma Pow˙is˙top: shows Pow(X) –is a topology″
⟨*proof*⟩

Empty set is open.

lemma empty˙open:
assumes T –is a topology″ shows 0 ∈ T
⟨*proof*⟩

The carrier is open.

lemma carr˙open: assumes T –is a topology″ shows ($\bigcup$T) ∈ T
⟨*proof*⟩

Union of a collection of open sets is open.

lemma union˙open: assumes T –is a topology″ and ∀A∈$\mathcal{A}$. A ∈ T
shows ($\bigcup\mathcal{A}$) ∈ T ⟨*proof*⟩

Union of a indexed family of open sets is open.

lemma union˙indexed˙open: assumes A1: T –is a topology″ and A2: ∀i∈I. P(i) ∈ T
shows ($\bigcup$i∈I. P(i)) ∈ T ⟨*proof*⟩

The intersection of any nonempty collection of topologies on a set X is a topology.

lemma Inter˙tops˙is˙top:
assumes A1: $\mathcal{M} \neq 0$ and A2: ∀T∈$\mathcal{M}$. T –is a topology″
shows ($\bigcap\mathcal{M}$) –is a topology″
⟨*proof*⟩

We will now introduce some notation. In Isar, this is done by definining a "locale". Locale is kind of a context that holds some assumptions and notation used in all theorems proven in it. In the locale (context) below called topology0 we assume that T is a topology. The interior of the set A (with respect to the topology in the context) is denoted int(A). The closure of a set $A \subseteq \bigcup T$ is denoted cl(A) and the boundary is ∂A.

locale topology0 =
fixes T
assumes topSpaceAssum: T –is a topology″

fixes int
defines int˙def [simp]: int(A) ≡ Interior(A,T)

fixes cl
defines cl˙def [simp]: cl(A) ≡ Closure(A,T)

fixes boundary (∂˙ [91] 92)
defines boundary˙def [simp]: ∂A ≡ Boundary(A,T)

Intersection of a finite nonempty collection of open sets is open.

lemma (in topology0) fin˙inter˙open˙open: assumes N≠0 N ∈ FinPow(T)
shows $\bigcap$N ∈ T
⟨*proof*⟩

Having a topology T and a set X we can define the induced topology as the one consisting of the intersections of X with sets from T. The notion of a collection restricted to a set is defined in ZF1.thy.

lemma (in topology0) Top˙1˙L4:
shows (T –restricted to″ X) –is a topology″
⟨*proof*⟩

50.2 Interior of a set

In this section we show basic properties of the interior of a set.

Interior of a set A is contained in A.

lemma (in topology0) Top_2_L1: shows int(A) ⊆ A
⟨*proof*⟩

Interior is open.

lemma (in topology0) Top_2_L2: shows int(A) ∈ T
⟨*proof*⟩

A set is open iff it is equal to its interior.

lemma (in topology0) Top_2_L3: shows U∈T ⟷ int(U) = U
⟨*proof*⟩

Interior of the interior is the interior.

lemma (in topology0) Top_2_L4: shows int(int(A)) = int(A)
⟨*proof*⟩

Interior of a bigger set is bigger.

lemma (in topology0) interior_mono:
assumes A1: A⊆B shows int(A) ⊆ int(B)
⟨*proof*⟩

An open subset of any set is a subset of the interior of that set.

lemma (in topology0) Top_2_L5: assumes U⊆A and U∈T
shows U ⊆ int(A)
⟨*proof*⟩

If a point of a set has an open neighboorhood contained in the set, then the point belongs to the interior of the set.

lemma (in topology0) Top_2_L6: assumes ∃U∈T. (x∈U ∧ U⊆A)
shows x ∈ int(A)
⟨*proof*⟩

A set is open iff its every point has a an open neighbourhood contained in the set. We will formulate this statement as two lemmas (implication one way and the other way). The lemma below shows that if a set is open then every point has a an open neighbourhood contained in the set.

lemma (in topology0) open_open_neigh:
assumes A1: V∈T
shows ∀x∈V. ∃U∈T. (x∈U ∧ U⊆V)
⟨*proof*⟩

If every point of a set has a an open neighbourhood contained in the set then the set is open.

lemma (in topology0) open˙neigh˙open:
  assumes A1: $\forall x\in V. \exists U\in T. (x\in U \wedge U\subseteq V)$
  shows $V\in T$
⟨*proof*⟩

The intersection of interiors is a equal to the interior of intersections.

lemma (in topology0) int˙inter˙int: shows $\text{int}(A) \cap \text{int}(B) = \text{int}(A\cap B)$
⟨*proof*⟩

50.3 Closed sets, closure, boundary.

This section is devoted to closed sets and properties of the closure and boundary operators.

The carrier of the space is closed.

lemma (in topology0) Top˙3˙L1: shows $(\bigcup T)$ –is closed in˝ T
⟨*proof*⟩

Empty set is closed.

lemma (in topology0) Top˙3˙L2: shows 0 –is closed in˝ T
  ⟨*proof*⟩

The collection of closed covers of a subset of the carrier of topology is never empty. This is good to know, as we want to intersect this collection to get the closure.

lemma (in topology0) Top˙3˙L3:
  assumes A1: $A \subseteq \bigcup T$ shows ClosedCovers(A,T) $\neq 0$
⟨*proof*⟩

Intersection of a nonempty family of closed sets is closed.

lemma (in topology0) Top˙3˙L4: assumes A1: $K\neq 0$ and
  A2: $\forall D\in K.$ D –is closed in˝ T
  shows $(\bigcap K)$ –is closed in˝ T
⟨*proof*⟩

The union and intersection of two closed sets are closed.

lemma (in topology0) Top˙3˙L5:
  assumes A1: D_1 –is closed in˝ T D_2 –is closed in˝ T
  shows
  $(D_1\cap D_2)$ –is closed in˝ T
  $(D_1\cup D_2)$ –is closed in˝ T
⟨*proof*⟩

Finite union of closed sets is closed. To understand the proof recall that $D \in$Pow$(\bigcup T)$ means that D is a subset of the carrier of the topology.

lemma (in topology0) fin˙union˙cl˙is˙cl:
  assumes

A1: N ∈ FinPow(–D∈Pow(⋃T). D –is closed in" T")
shows (⋃N) –is closed in" T
⟨*proof*⟩

Closure of a set is closed.

lemma (in topology0) cl˙is˙closed: assumes A ⊆ ⋃T
shows cl(A) –is closed in" T
⟨*proof*⟩

Closure of a bigger sets is bigger.

lemma (in topology0) top˙closure˙mono:
assumes A1: A ⊆ ⋃T B ⊆ ⋃T and A2:A⊆B
shows cl(A) ⊆ cl(B)
⟨*proof*⟩

Boundary of a set is closed.

lemma (in topology0) boundary˙closed:
assumes A1: A ⊆ ⋃T shows ∂A –is closed in" T
⟨*proof*⟩

A set is closed iff it is equal to its closure.

lemma (in topology0) Top˙3˙L8: assumes A1: A ⊆ ⋃T
shows A –is closed in" T ⟷ cl(A) = A
⟨*proof*⟩

Complement of an open set is closed.

lemma (in topology0) Top˙3˙L9:
assumes A1: A∈T
shows (⋃T - A) –is closed in" T
⟨*proof*⟩

A set is contained in its closure.

lemma (in topology0) cl˙contains˙set: assumes A ⊆ ⋃T shows A ⊆ cl(A)
⟨*proof*⟩

Closure of a subset of the carrier is a subset of the carrier and closure of the complement is the complement of the interior.

lemma (in topology0) Top˙3˙L11: assumes A1: A ⊆ ⋃T
shows
cl(A) ⊆ ⋃T
cl(⋃T - A) = ⋃T - int(A)
⟨*proof*⟩

Boundary of a set is the closure of the set minus the interior of the set.

lemma (in topology0) Top˙3˙L12: assumes A1: A ⊆ ⋃T
shows ∂A = cl(A) - int(A)
⟨*proof*⟩

If a set A is contained in a closed set B, then the closure of A is contained in B.

lemma (in topology0) Top˙3˙L13:
  assumes A1: B –is closed in" T   A⊆B
  shows cl(A) ⊆ B
⟨*proof*⟩

If a set is disjoint with an open set, then we can close it and it will still be disjoint.

lemma (in topology0) disj˙open˙cl˙disj:
  assumes A1: A ⊆ ⋃T V∈T and A2: A∩V = 0
  shows cl(A) ∩ V = 0
⟨*proof*⟩

A reformulation of disj˙open˙cl˙disj: If a point belongs to the closure of a set, then we can find a point from the set in any open neighboorhood of the point.

lemma (in topology0) cl˙inter˙neigh:
  assumes A ⊆ ⋃T and U∈T and x ∈ cl(A) ∩ U
  shows A∩U ≠ 0 ⟨*proof*⟩

A reverse of cl˙inter˙neigh: if every open neiboorhood of a point has a nonempty intersection with a set, then that point belongs to the closure of the set.

lemma (in topology0) inter˙neigh˙cl:
  assumes A1: A ⊆ ⋃T and A2: x∈⋃T and A3: ∀U∈T. x∈U ⟶ U∩A ≠ 0
  shows x ∈ cl(A)
⟨*proof*⟩

end

51 Topology 1

theory Topology˙ZF˙1 imports Topology˙ZF

begin

In this theory file we study separation axioms and the notion of base and subbase. Using the products of open sets as a subbase we define a natural topology on a product of two topological spaces.

51.1 Separation axioms.

Topological spaces cas be classified according to certain properties called "separation axioms". In this section we define what it means that a topological space is T_0, T_1 or T_2.

A topology on X is T_0 if for every pair of distinct points of X there is an open set that contains only one of them.

definition
  isT0 (˙ –is T_0″ [90] 91) where
  T –is T_0″ ≡ ∀ x y. ((x ∈ ⋃T ∧ y ∈ ⋃T ∧ x≠y) ⟶
  (∃U∈T. (x∈U ∧ y∉U) ∨ (y∈U ∧ x∉U)))

A topology is T_1 if for every such pair there exist an open set that contains the first point but not the second.

definition
  isT1 (˙ –is T_1″ [90] 91) where
  T –is T_1″ ≡ ∀ x y. ((x ∈ ⋃T ∧ y ∈ ⋃T ∧ x≠y) ⟶
  (∃U∈T. (x∈U ∧ y∉U)))

A topology is T_2 (Hausdorff) if for every pair of points there exist a pair of disjoint open sets each containing one of the points. This is an important class of topological spaces. In particular, metric spaces are Hausdorff.

definition
  isT2 (˙ –is T_2″ [90] 91) where
  T –is T_2″ ≡ ∀ x y. ((x ∈ ⋃T ∧ y ∈ ⋃T ∧ x≠y) ⟶
  (∃U∈T. ∃V∈T. x∈U ∧ y∈V ∧ U∩V=0))

If a topology is T_1 then it is T_0. We don't really assume here that T is a topology on X. Instead, we prove the relation between isT0 condition and isT1.

lemma T1˙is˙T0: assumes A1: T –is T_1″ shows T –is T_0″
⟨*proof*⟩

If a topology is T_2 then it is T_1.

lemma T2˙is˙T1: assumes A1: T –is T_2″ shows T –is T_1″
⟨*proof*⟩

In a T_0 space two points that can not be separated by an open set are equal. Proof by contradiction.

lemma Top˙1˙1˙L1: assumes A1: T –is T_0″ and A2: x ∈ ⋃T y ∈ ⋃T
  and A3: ∀U∈T. (x∈U ⟷ y∈U)
  shows x=y
⟨*proof*⟩

51.2 Bases and subbases.

Sometimes it is convenient to talk about topologies in terms of their bases and subbases. These are certain collections of open sets that define the whole topology.

A base of topology is a collection of open sets such that every open set is a union of the sets from the base.

definition
IsAbaseFor (infixl –is a base for″ 65) where
B –is a base for″ T ≡ B⊆T ∧ T = –⋃A. A∈Pow(B)″

A subbase is a collection of open sets such that finite intersection of those sets form a base.

definition
IsAsubBaseFor (infixl –is a subbase for″ 65) where
B –is a subbase for″ T ≡
B ⊆ T ∧ –⋂A. A ∈ FinPow(B)″ –is a base for″ T

Below we formulate a condition that we will prove to be necessary and sufficient for a collection B of open sets to form a base. It says that for any two sets U, V from the collection B we can find a point $x \in U \cap V$ with a neighboorhod from B contained in $U \cap V$.

definition
SatisfiesBaseCondition (˙ –satisfies the base condition″ [50] 50)
where
B –satisfies the base condition″ ≡
∀U V. ((U∈B ∧ V∈B) ⟶ (∀x ∈ U∩V. ∃W∈B. x∈W ∧ W ⊆ U∩V))

A collection that is closed with respect to intersection satisfies the base condition.

lemma inter˙closed˙base: assumes ∀U∈B.(∀V∈B. U∩V ∈ B)
shows B –satisfies the base condition″
⟨*proof*⟩

Each open set is a union of some sets from the base.

lemma Top˙1˙2˙L1: assumes B –is a base for″ T and U∈T
shows ∃A∈Pow(B). U = ⋃A
⟨*proof*⟩

Elements of base are open.

lemma base˙sets˙open:
assumes B –is a base for″ T and U ∈ B
shows U ∈ T
⟨*proof*⟩

A base defines topology uniquely.

lemma same˙base˙same˙top:
assumes B –is a base for″ T and B –is a base for″ S
shows T = S
⟨*proof*⟩

Every point from an open set has a neighboorhood from the base that is contained in the set.

lemma point˙open˙base˙neigh:

assumes A1: B –is a base for″ T and A2: U∈T and A3: x∈U
shows ∃ V∈B. V⊆U ∧ x∈V
⟨*proof*⟩

A criterion for a collection to be a base for a topology that is a slight reformulation of the definition. The only thing different that in the definition is that we assume only that every open set is a union of some sets from the base. The definition requires also the opposite inclusion that every union of the sets from the base is open, but that we can prove if we assume that T is a topology.

lemma is˙a˙base˙criterion: assumes A1: T –is a topology″
and A2: B ⊆ T and A3: ∀ V ∈ T. ∃ A ∈ Pow(B). V = ⋃ A
shows B –is a base for″ T
⟨*proof*⟩

A necessary condition for a collection of sets to be a base for some topology : every point in the intersection of two sets in the base has a neighboorhood from the base contained in the intersection.

lemma Top˙1˙2˙L2:
assumes A1:∃ T. T –is a topology″ ∧ B –is a base for″ T
and A2: V∈B W∈B
shows ∀ x ∈ V∩W. ∃ U∈B. x∈U ∧ U ⊆ V ∩ W
⟨*proof*⟩

We will construct a topology as the collection of unions of (would-be) base. First we prove that if the collection of sets satisfies the condition we want to show to be sufficient, the the intersection belongs to what we will define as topology (am I clear here?). Having this fact ready simplifies the proof of the next lemma. There is not much topology here, just some set theory.

lemma Top˙1˙2˙L3:
assumes A1: ∀x∈ V∩W . ∃ U∈B. x∈U ∧ U ⊆ V∩W
shows V∩W ∈ –⋃ A. A∈Pow(B)″
⟨*proof*⟩

The next lemma is needed when proving that the would-be topology is closed with respect to taking intersections. We show here that intersection of two sets from this (would-be) topology can be written as union of sets from the topology.

lemma Top˙1˙2˙L4:
assumes A1: U_1 ∈ –⋃ A. A∈Pow(B)″ U_2 ∈ –⋃ A. A∈Pow(B)″
and A2: B –satisfies the base condition″
shows ∃ C. C ⊆ –⋃ A. A∈Pow(B)″ ∧ U_1∩U_2 = ⋃ C
⟨*proof*⟩

If B satisfies the base condition, then the collection of unions of sets from B is a topology and B is a base for this topology.

theorem Top˙1˙2˙T1:
  assumes A1: B –satisfies the base condition˝
  and A2: T = –⋃A. A∈Pow(B)˝
  shows T –is a topology˝ and B –is a base for˝ T
⟨*proof*⟩

The carrier of the base and topology are the same.

lemma Top˙1˙2˙L5: assumes B –is a base for˝ T
  shows ⋃T = ⋃B
  ⟨*proof*⟩

If B is a base for T, then T is the smallest topology containing B.

lemma base˙smallest˙top:
  assumes A1: B –is a base for˝ T and A2: S –is a topology˝ and A3: B⊆S
  shows T⊆S
⟨*proof*⟩

If B is a base for T and B is a topology, then $B = T$.

lemma base˙topology: assumes B –is a topology˝ and B –is a base for˝ T
  shows B=T ⟨*proof*⟩

51.3 Product topology

In this section we consider a topology defined on a product of two sets.

Given two topological spaces we can define a topology on the product of the carriers such that the cartesian products of the sets of the topologies are a base for the product topology. Recall that for two collections S, T of sets the product collection is defined (in ZF1.thy) as the collections of cartesian products $A \times B$, where $A \in S, B \in T$.

definition
  ProductTopology(T,S) ≡ –⋃W. W ∈ Pow(ProductCollection(T,S))˝

The product collection satisfies the base condition.

lemma Top˙1˙4˙L1:
  assumes A1: T –is a topology˝ S –is a topology˝
  and A2: A ∈ ProductCollection(T,S) B ∈ ProductCollection(T,S)
  shows ∀x∈(A∩B). ∃W∈ProductCollection(T,S). (x∈W ∧ W ⊆ A ∩ B)
⟨*proof*⟩

The product topology is indeed a topology on the product.

theorem Top˙1˙4˙T1: assumes A1: T –is a topology˝ S –is a topology˝
  shows
  ProductTopology(T,S) –is a topology˝
  ProductCollection(T,S) –is a base for˝ ProductTopology(T,S)
  ⋃ ProductTopology(T,S) = ⋃T × ⋃S
⟨*proof*⟩

Each point of a set open in the product topology has a neighborhood which is a cartesian product of open sets.

lemma prod˙top˙point˙neighb:
  assumes A1: T –is a topology˝ S –is a topology˝ and
  A2: U ∈ ProductTopology(T,S) and A3: x ∈ U
  shows ∃V W. V∈T ∧ W∈S ∧ V×W ⊆ U ∧ x ∈ V×W
⟨*proof*⟩

Products of open sets are open in the product topology.

lemma prod˙open˙open˙prod:
  assumes A1: T –is a topology˝ S –is a topology˝ and
  A2: U∈T V∈S
  shows U×V ∈ ProductTopology(T,S)
⟨*proof*⟩

Sets that are open in th product topology are contained in the product of the carrier.

lemma prod˙open˙type: assumes A1: T –is a topology˝ S –is a topology˝ and
  A2: V ∈ ProductTopology(T,S)
  shows V ⊆ ⋃T × ⋃S
⟨*proof*⟩

Suppose we have subsets $A \subseteq X, B \subseteq Y$, where X, Y are topological spaces with topologies T, S. We can the consider relative topologies on T_A, S_B on sets A, B and the collection of cartesian products of sets open in T_A, S_B, (namely $\{U \times V : U \in T_A, V \in S_B\}$. The next lemma states that this collection is a base of the product topology on $X \times Y$ restricted to the product $A \times B$.

lemma prod˙restr˙base˙restr:
  assumes A1: T –is a topology˝ S –is a topology˝
  shows
  ProductCollection(T –restricted to˝ A, S –restricted to˝ B)
  –is a base for˝ (ProductTopology(T,S) –restricted to˝ A×B)
⟨*proof*⟩

We can commute taking restriction (relative topology) and product topology. The reason the two topologies are the same is that they have the same base.

lemma prod˙top˙restr˙comm:
  assumes A1: T –is a topology˝ S –is a topology˝
  shows
  ProductTopology(T –restricted to˝ A,S –restricted to˝ B) =
  ProductTopology(T,S) –restricted to˝ (A×B)
⟨*proof*⟩

Projection of a section of an open set is open.

lemma prod˙sec˙open1: assumes A1: T –is a topology˝ S –is a topology˝ and
  A2: V ∈ ProductTopology(T,S) and A3: x ∈ ⋃T

shows {y ∈ ⋃S. ⟨x,y⟩ ∈ V} ∈ S
⟨*proof*⟩

Projection of a section of an open set is open. This is dual of prod_sec_open1 with a very similar proof.

lemma prod_sec_open2: assumes A1: T {is a topology} S {is a topology} and
A2: V ∈ ProductTopology(T,S) and A3: y ∈ ⋃S
shows {x ∈ ⋃T. ⟨x,y⟩ ∈ V} ∈ T
⟨*proof*⟩

end

52 Topology 1b

theory Topology_ZF_1b imports Topology_ZF_1

begin

One of the facts demonstrated in every class on General Topology is that in a T_2 (Hausdorff) topological space compact sets are closed. Formalizing the proof of this fact gave me an interesting insight into the role of the Axiom of Choice (AC) in many informal proofs.

A typical informal proof of this fact goes like this: we want to show that the complement of K is open. To do this, choose an arbitrary point $y \in K^c$. Since X is T_2, for every point $x \in K$ we can find an open set U_x such that $y \notin \overline{U_x}$. Obviously $\{U_x\}_{x\in K}$ covers K, so select a finite subcollection that covers K, and so on. I had never realized that such reasoning requires the Axiom of Choice. Namely, suppose we have a lemma that states "In T_2 spaces, if $x \neq y$, then there is an open set U such that $x \in U$ and $y \notin \overline{U}$" (like our lemma T2_cl_open_sep below). This only states that the set of such open sets U is not empty. To get the collection $\{U_x\}_{x\in K}$ in this proof we have to select one such set among many for every $x \in K$ and this is where we use the Axiom of Choice. Probably in 99/100 cases when an informal calculus proof states something like $\forall \varepsilon \exists \delta_\varepsilon \cdots$ the proof uses AC. Most of the time the use of AC in such proofs can be avoided. This is also the case for the fact that in a T_2 space compact sets are closed.

52.1 Compact sets are closed - no need for AC

In this section we show that in a T_2 topological space compact sets are closed.

First we prove a lemma that in a T_2 space two points can be separated by the closure of an open set.

lemma (in topology0) T2˙cl˙open˙sep:
  assumes T –is T_2˝ and x ∈ ⋃T y ∈ ⋃T x≠y
  shows ∃U∈T. (x∈U ∧ y ∉ cl(U))
⟨*proof*⟩

AC-free proof that in a Hausdorff space compact sets are closed. To understand the notation recall that in Isabelle/ZF Pow(A) is the powerset (the set of subsets) of A and FinPow(A) denotes the set of finite subsets of A in IsarMathLib.

theorem (in topology0) in˙t2˙compact˙is˙cl:
  assumes A1: T –is T_2˝ and A2: K –is compact in˝ T
  shows K –is closed in˝ T
⟨*proof*⟩

end

53 Topology 2

theory Topology˙ZF˙2 imports Topology˙ZF˙1 func1 Fol1

begin

This theory continues the series on general topology and covers the definition and basic properties of continuous functions. We also introduce the notion of homeomorphism an prove the pasting lemma.

53.1 Continuous functions.

In this section we define continuous functions and prove that certain conditions are equivalent to a function being continuous.

In standard math we say that a function is contiuous with respect to two topologies τ_1, τ_2 if the inverse image of sets from topology τ_2 are in τ_1. Here we define a predicate that is supposed to reflect that definition, with a difference that we don't require in the definition that τ_1, τ_2 are topologies. This means for example that when we define measurable functions, the definition will be the same.
The notation f-(A) means the inverse image of (a set) A with respect to (a function) f.

definition
  IsContinuous(τ_1,τ_2,f) ≡ (∀U∈τ_2. f-(U) ∈ τ_1)

A trivial example of a continuous function - identity is continuous.

lemma id˙cont: shows IsContinuous(τ,τ,id(⋃τ))
⟨*proof*⟩

We will work with a pair of topological spaces. The following locale sets up our context that consists of two topologies τ_1, τ_2 and a continuous function $f : X_1 \to X_2$, where X_i is defined as $\bigcup \tau_i$ for $i = 1, 2$. We also define notation cl_1(A) and cl_2(A) for closure of a set A in topologies τ_1 and τ_2, respectively.

locale two˙top˙spaces0 =

fixes τ_1
assumes tau1˙is˙top: τ_1 –is a topology˝

fixes τ_2
assumes tau2˙is˙top: τ_2 –is a topology˝

fixes X_1
defines X1˙def [simp]: $X_1 \equiv \bigcup \tau_1$

fixes X_2
defines X2˙def [simp]: $X_2 \equiv \bigcup \tau_2$

fixes f
assumes fmapAssum: f: $X_1 \to X_2$

fixes isContinuous (˙ –is continuous˝ [50] 50)
defines isContinuous˙def [simp]: g –is continuous˝ $\equiv$ IsContinuous(τ_1,τ_2,g)

fixes cl_1
defines cl1˙def [simp]: cl_1(A) $\equiv$ Closure(A,τ_1)

fixes cl_2
defines cl2˙def [simp]: cl_2(A) $\equiv$ Closure(A,τ_2)

First we show that theorems proven in locale topology0 are valid when applied to topologies τ_1 and τ_2.

lemma (in two˙top˙spaces0) topol˙cntxs˙valid:
shows topology0(τ_1) and topology0(τ_2)
⟨*proof*⟩

For continuous functions the inverse image of a closed set is closed.

lemma (in two˙top˙spaces0) TopZF˙2˙1˙L1:
assumes A1: f –is continuous˝ and A2: D –is closed in˝ τ_2
shows f-(D) –is closed in˝ τ_1
⟨*proof*⟩

If the inverse image of every closed set is closed, then the image of a closure is contained in the closure of the image.

lemma (in two˙top˙spaces0) Top˙ZF˙2˙1˙L2:
assumes A1: $\forall$D. ((D –is closed in˝ τ_2) $\longrightarrow$ f-(D) –is closed in˝ τ_1)
and A2: A $\subseteq$ X_1
shows f(cl_1(A)) $\subseteq$ cl_2(f(A))

⟨*proof*⟩

If $f\left(\overline{A}\right) \subseteq \overline{f(A)}$ (the image of the closure is contained in the closure of the image), then $\overline{f^{-1}(B)} \subseteq f^{-1}\left(\overline{B}\right)$ (the inverse image of the closure contains the closure of the inverse image).

lemma (in two˙top˙spaces0) Top˙ZF˙2˙1˙L3:
  assumes A1: ∀ A. (A ⊆ X_1 ⟶ f(cl_1(A)) ⊆ cl_2(f(A)))
  shows ∀B. (B ⊆ X_2 ⟶ cl_1(f-(B)) ⊆ f-(cl_2(B)))
⟨*proof*⟩

If $\overline{f^{-1}(B)} \subseteq f^{-1}\left(\overline{B}\right)$ (the inverse image of a closure contains the closure of the inverse image), then the function is continuous. This lemma closes a series of implications in lemmas Top˙ZF˙2˙1˙L1, Top˙ZF˙2˙1˙L2 and Top˙ZF˙2˙1˙L3 showing equivalence of four definitions of continuity.

lemma (in two˙top˙spaces0) Top˙ZF˙2˙1˙L4:
  assumes A1: ∀B. (B ⊆ X_2 ⟶ cl_1(f-(B)) ⊆ f-(cl_2(B)))
  shows f –is continuous″
⟨*proof*⟩

Another condition for continuity: it is sufficient to check if the inverse image of every set in a base is open.

lemma (in two˙top˙spaces0) Top˙ZF˙2˙1˙L5:
  assumes A1: B –is a base for″ τ_2 and A2: ∀U∈B. f-(U) ∈ τ_1
  shows f –is continuous″
⟨*proof*⟩

We can strenghten the previous lemma: it is sufficient to check if the inverse image of every set in a subbase is open. The proof is rather awkward, as usual when we deal with general intersections. We have to keep track of the case when the collection is empty.

lemma (in two˙top˙spaces0) Top˙ZF˙2˙1˙L6:
  assumes A1: B –is a subbase for″ τ_2 and A2: ∀U∈B. f-(U) ∈ τ_1
  shows f –is continuous″
⟨*proof*⟩

A dual of Top˙ZF˙2˙1˙L5: a function that maps base sets to open sets is open.

lemma (in two˙top˙spaces0) base˙image˙open:
  assumes A1: $\mathcal{B}$ –is a base for″ τ_1 and A2: ∀B∈$\mathcal{B}$. f(B) ∈ τ_2 and A3: U∈τ_1
  shows f(U) ∈ τ_2
⟨*proof*⟩

A composition of two continuous functions is continuous.

lemma comp˙cont: assumes IsContinuous(T,S,f) and IsContinuous(S,R,g)
  shows IsContinuous(T,R,g O f)
  ⟨*proof*⟩

A composition of three continuous functions is continuous.

lemma comp˙cont3:
assumes IsContinuous(T,S,f) and IsContinuous(S,R,g) and IsContinuous(R,P,h)
shows IsContinuous(T,P,h O g O f)
⟨*proof*⟩

53.2 Homeomorphisms

This section studies "homeomorphisms" - continous bijections whose inverses are also continuous. Notions that are preserved by (commute with) homeomorphisms are called "topological invariants".

Homeomorphism is a bijection that preserves open sets.

definition IsAhomeomorphism(T,S,f) ≡
f ∈ bij(⋃T,⋃S) ∧ IsContinuous(T,S,f) ∧ IsContinuous(S,T,converse(f))

Inverse (converse) of a homeomorphism is a homeomorphism.

lemma homeo˙inv: assumes IsAhomeomorphism(T,S,f)
shows IsAhomeomorphism(S,T,converse(f))
⟨*proof*⟩

Homeomorphisms are open maps.

lemma homeo˙open: assumes IsAhomeomorphism(T,S,f) and U∈T
shows f(U) ∈ S
⟨*proof*⟩

A continuous bijection that is an open map is a homeomorphism.

lemma bij˙cont˙open˙homeo:
assumes f ∈ bij(⋃T,⋃S) and IsContinuous(T,S,f) and ∀U∈T. f(U) ∈ S
shows IsAhomeomorphism(T,S,f)
⟨*proof*⟩

A continuous bijection that maps base to open sets is a homeomorphism.

lemma (in two˙top˙spaces0) bij˙base˙open˙homeo:
assumes A1: f ∈ bij(X_1,X_2) and A2: $\mathcal{B}$ –is a base for" τ_1 and A3: $\mathcal{C}$ –is a base for" τ_2 and
A4: ∀U∈$\mathcal{C}$. f-(U) ∈ τ_1 and A5: ∀V∈$\mathcal{B}$. f(V) ∈ τ_2
shows IsAhomeomorphism(τ_1,τ_2,f)
⟨*proof*⟩

A bijection that maps base to base is a homeomorphism.

lemma (in two˙top˙spaces0) bij˙base˙homeo:
assumes A1: f ∈ bij(X_1,X_2) and A2: $\mathcal{B}$ –is a base for" τ_1 and
A3: –f(B). B∈$\mathcal{B}$" –is a base for" τ_2
shows IsAhomeomorphism(τ_1,τ_2,f)
⟨*proof*⟩

Interior is a topological invariant.

theorem int˙top˙invariant: assumes A1: A⊆⋃T and A2: IsAhomeomorphism(T,S,f)
shows f(Interior(A,T)) = Interior(f(A),S)
⟨*proof*⟩

53.3 Topologies induced by mappings

In this section we consider various ways a topology may be defined on a set that is the range (or the domain) of a function whose domain (or range) is a topological space.

A bijection from a topological space induces a topology on the range.

theorem bij˙induced˙top: assumes A1: T –is a topology˝ and A2: f ∈ bij(⋃T,Y)
shows
–f(U). U∈T˝ –is a topology˝ and
– –f(x).x∈U˝. U∈T˝ –is a topology˝ and
(⋃–f(U). U∈T˝) = Y and
IsAhomeomorphism(T, –f(U). U∈T˝,f)
⟨*proof*⟩

53.4 Partial functions and continuity

Suppose we have two topologies τ_1, τ_2 on sets $X_i = \bigcup \tau_i, i = 1, 2$. Consider some function $f : A \to X_2$, where $A \subseteq X_1$ (we will call such function "partial"). In such situation we have two natural possibilities for the pairs of topologies with respect to which this function may be continuous. One is obviously the original τ_1, τ_2 and in the second one the first element of the pair is the topology relative to the domain of the function: $\{A \cap U | U \in \tau_1\}$. These two possibilities are not exactly the same and the goal of this section is to explore the differences.

If a function is continuous, then its restriction is continous in relative topology.

lemma (in two˙top˙spaces0) restr˙cont:
assumes A1: A ⊆ X_1 and A2: f –is continuous˝
shows IsContinuous(τ_1 –restricted to˝ A, τ_2,restrict(f,A))
⟨*proof*⟩

If a function is continuous, then it is continuous when we restrict the topology on the range to the image of the domain.

lemma (in two˙top˙spaces0) restr˙image˙cont:
assumes A1: f –is continuous˝
shows IsContinuous(τ_1, τ_2 –restricted to˝ f(X_1),f)
⟨*proof*⟩

A combination of restr˙cont and restr˙image˙cont.

lemma (in two˙top˙spaces0) restr˙restr˙image˙cont:

assumes A1: A $\subseteq$ X_1 and A2: f –is continuous" and
A3: g = restrict(f,A) and
A4: τ_3 = τ_1 –restricted to" A
shows IsContinuous(τ_3, τ_2 –restricted to" g(A),g)
⟨*proof*⟩

We need a context similar to two˙top˙spaces0 but without the global function $f : X_1 \to X_2$.

locale two˙top˙spaces1 =

fixes τ_1
assumes tau1˙is˙top: τ_1 –is a topology"

fixes τ_2
assumes tau2˙is˙top: τ_2 –is a topology"

fixes X_1
defines X1˙def [simp]: $X_1 \equiv \bigcup \tau_1$

fixes X_2
defines X2˙def [simp]: $X_2 \equiv \bigcup \tau_2$

If a partial function $g : X_1 \supseteq A \to X_2$ is continuous with respect to (τ_1, τ_2), then A is open (in τ_1) and the function is continuous in the relative topology.

lemma (in two˙top˙spaces1) partial˙fun˙cont:
assumes A1: g:A→X_2 and A2: IsContinuous(τ_1,τ_2,g)
shows A $\in$ τ_1 and IsContinuous(τ_1 –restricted to" A, τ_2, g)
⟨*proof*⟩

For partial function defined on open sets continuity in the whole and relative topologies are the same.

lemma (in two˙top˙spaces1) part˙fun˙on˙open˙cont:
assumes A1: g:A→X_2 and A2: A $\in$ τ_1
shows IsContinuous(τ_1,τ_2,g) $\longleftrightarrow$
IsContinuous(τ_1 –restricted to" A, τ_2, g)
⟨*proof*⟩

53.5 Product topology and continuity

We start with three topological spaces $(\tau_1, X_1), (\tau_2, X_2)$ and (τ_3, X_3) and a function $f : X_1 \times X_2 \to X_3$. We will study the properties of f with respect to the product topology $\tau_1 \times \tau_2$ and τ_3. This situation is similar as in locale two˙top˙spaces0 but the first topological space is assumed to be a product of two topological spaces.

First we define a locale with three topological spaces.

locale prod˙top˙spaces0 =

fixes τ_1
assumes tau1˙is˙top: τ_1 –is a topology″

fixes τ_2
assumes tau2˙is˙top: τ_2 –is a topology″

fixes τ_3
assumes tau3˙is˙top: τ_3 –is a topology″

fixes X_1
defines X1˙def [simp]: $X_1 \equiv \bigcup \tau_1$

fixes X_2
defines X2˙def [simp]: $X_2 \equiv \bigcup \tau_2$

fixes X_3
defines X3˙def [simp]: $X_3 \equiv \bigcup \tau_3$

fixes η
defines eta˙def [simp]: $\eta \equiv$ ProductTopology(τ_1,τ_2)

Fixing the first variable in a two-variable continuous function results in a continuous function.

lemma (in prod˙top˙spaces0) fix˙1st˙var˙cont:
assumes f: $X_1 \times X_2 \rightarrow X_3$ and IsContinuous(η,τ_3,f)
and $x \in X_1$
shows IsContinuous(τ_2,τ_3,Fix1stVar(f,x))
⟨*proof*⟩

Fixing the second variable in a two-variable continuous function results in a continuous function.

lemma (in prod˙top˙spaces0) fix˙2nd˙var˙cont:
assumes f: $X_1 \times X_2 \rightarrow X_3$ and IsContinuous(η,τ_3,f)
and $y \in X_2$
shows IsContinuous(τ_1,τ_3,Fix2ndVar(f,y))
⟨*proof*⟩

Having two constinuous mappings we can construct a third one on the cartesian product of the domains.

lemma cart˙prod˙cont:
assumes A1: τ_1 –is a topology″ τ_2 –is a topology″ and
A2: η_1 –is a topology″ η_2 –is a topology″ and
A3a: $f_1:\bigcup \tau_1 \rightarrow \bigcup \eta_1$ and A3b: $f_2:\bigcup \tau_2 \rightarrow \bigcup \eta_2$ and
A4: IsContinuous(τ_1,η_1,f_1) IsContinuous(τ_2,η_2,f_2) and
A5: g = –$\langle p,\langle f_1(fst(p)),f_2(snd(p))\rangle\rangle$. $p \in \bigcup \tau_1 \times \bigcup \tau_2$″
shows IsContinuous(ProductTopology(τ_1,τ_2),ProductTopology(η_1,η_2),g)
⟨*proof*⟩

A reformulation of the cart˙prod˙cont lemma above in slightly different notation.

theorem (in two˙top˙spaces0) product˙cont˙functions:
 assumes f:$X_1 \rightarrow X_2$ g:$\bigcup\tau_3 \rightarrow \bigcup\tau_4$
 IsContinuous(τ_1,τ_2,f) IsContinuous(τ_3,τ_4,g)
 τ_4–is a topology″ τ_3–is a topology″
 shows IsContinuous(ProductTopology(τ_1,τ_3),ProductTopology(τ_2,τ_4),–$\langle\langle$x,y$\rangle$,$\langle$fx,gy$\rangle\rangle$. $\langle$x,y$\rangle \in X_1 \times \bigcup\tau_3$″)
$\langle$*proof*$\rangle$

A special case of cart˙prod˙cont when the function acting on the second axis is the identity.

lemma cart˙prod˙cont1:
 assumes A1: τ_1 –is a topology″ and A1a: τ_2 –is a topology″ and
 A2: η_1 –is a topology″ and
 A3: f_1:$\bigcup\tau_1 \rightarrow \bigcup\eta_1$ and A4: IsContinuous(τ_1,η_1,f_1) and
 A5: g = –$\langle$p, $\langle f_1$(fst(p)),snd(p)$\rangle\rangle$. p $\in \bigcup\tau_1 \times \bigcup\tau_2$″
 shows IsContinuous(ProductTopology(τ_1,τ_2),ProductTopology(η_1,τ_2),g)
$\langle$*proof*$\rangle$

53.6 Pasting lemma

The classical pasting lemma states that if U_1, U_2 are both open (or closed) and a function is continuous when restricted to both U_1 and U_2 then it is continuous when restricted to $U_1 \cup U_2$. In this section we prove a generalization statement stating that the set $\{U \in \tau_1 | f|_U$ is continuous $\}$ is a topology.

A typical statement of the pasting lemma uses the notion of a function restricted to a set being continuous without specifying the topologies with respect to which this continuity holds. In two˙top˙spaces0 context the notation g –is continuous″ means continuity wth respect to topologies τ_1, τ_2. The next lemma is a special case of partial˙fun˙cont and states that if for some set $A \subseteq X_1 = \bigcup \tau_1$ the function $f|_A$ is continuous (with respect to (τ_1, τ_2)), then A has to be open. This clears up terminology and indicates why we need to pay attention to the issue of which topologies we talk about when we say that the restricted (to some closed set for example) function is continuos.

lemma (in two˙top˙spaces0) restriction˙continuous1:
 assumes A1: A $\subseteq X_1$ and A2: restrict(f,A) –is continuous″
 shows A $\in \tau_1$
$\langle$*proof*$\rangle$

If a fuction is continuous on each set of a collection of open sets, then it is continuous on the union of them. We could use continuity with respect to the relative topology here, but we know that on open sets this is the same as the original topology.

lemma (in two˙top˙spaces0) pasting˙lemma1:
  assumes A1: M $\subseteq$ τ_1 and A2: $\forall$U$\in$M. restrict(f,U) –is continuous″
  shows restrict(f,$\bigcup$M) –is continuous″
⟨*proof*⟩

If a function is continuous on two sets, then it is continuous on intersection.

lemma (in two˙top˙spaces0) cont˙inter˙cont:
  assumes A1: A $\subseteq$ X_1 B $\subseteq$ X_1 and
  A2: restrict(f,A) –is continuous″ restrict(f,B) –is continuous″
  shows restrict(f,A$\cap$B) –is continuous″
⟨*proof*⟩

The collection of open sets U such that f restricted to U is continuous, is a topology.

theorem (in two˙top˙spaces0) pasting˙theorem:
  shows –U $\in$ τ_1. restrict(f,U) –is continuous″″ –is a topology″
⟨*proof*⟩

0 is continuous.

corollary (in two˙top˙spaces0) zero˙continuous: shows 0 –is continuous″
⟨*proof*⟩

end

54 Topology 3

theory Topology˙ZF˙3 imports Topology˙ZF˙2 FiniteSeq˙ZF

begin

Topology˙ZF˙1 theory describes how we can define a topology on a product of two topological spaces. One way to generalize that is to construct topology for a cartesian product of n topological spaces. The cartesian product approach is somewhat inconvenient though. Another way to approach product topology on X^n is to model cartesian product as sets of sequences (of length n) of elements of X. This means that having a topology on X we want to define a toplogy on the space $n \to X$, where n is a natural number (recall that $n = \{0, 1, ..., n-1\}$ in ZF). However, this in turn can be done more generally by defining a topology on any function space $I \to X$, where I is any set of indices. This is what we do in this theory.

54.1 The base of the product topology

In this section we define the base of the product topology.

Suppose $\mathcal{X} = I \to \bigcup T$ is a space of functions from some index set I to the carrier of a topology T. Then take a finite collection of open sets $W : N \to T$

indexed by $N \subseteq I$. We can define a subset of $\mathcal{X}$ that models the cartesian product of W.

definition
  FinProd($\mathcal{X}$,W) ≡ –x∈$\mathcal{X}$. ∀ i∈domain(W). x(i) ∈ W(i)˝

Now we define the base of the product topology as the collection of all finite products (in the sense defined above) of open sets.

definition
  ProductTopBase(I,T) ≡ ⋃N∈FinPow(I).–FinProd(I→⋃T,W). W∈N→T˝

Finally, we define the product topology on sequences. We use the "Seq" prefix although the definition is good for any index sets, not only natural numbers.

definition
  SeqProductTopology(I,T) ≡ –⋃B. B ∈ Pow(ProductTopBase(I,T))˝

Product topology base is closed with respect to intersections.

lemma prod˙top˙base˙inter:
  assumes A1: T –is a topology˝ and
  A2: U ∈ ProductTopBase(I,T) V ∈ ProductTopBase(I,T)
  shows U∩V ∈ ProductTopBase(I,T)
⟨*proof*⟩

In the next theorem we show the collection of sets defined above as ProductTopBase($\mathcal{X}$,T) satisfies the base condition. This is a condition, defined in Topology˙ZF˙1 that allows to claim that this collection is a base for some topology.

theorem prod˙top˙base˙is˙base: assumes T –is a topology˝
  shows ProductTopBase(I,T) –satisfies the base condition˝
  ⟨*proof*⟩

The (sequence) product topology is indeed a topology on the space of sequences. In the proof we are using the fact that $(\emptyset \to X) = \{\emptyset\}$.

theorem seq˙prod˙top˙is˙top: assumes T –is a topology˝
  shows
  SeqProductTopology(I,T) –is a topology˝ and
  ProductTopBase(I,T) –is a base for˝ SeqProductTopology(I,T) and
  ⋃SeqProductTopology(I,T) = (I→⋃T)
⟨*proof*⟩

54.2 Finite product of topologies

As a special case of the space of functions $I \to X$ we can consider space of lists of elements of X, i.e. space $n \to X$, where n is a natural number (recall that in ZF set theory $n = \{0, 1, ..., n-1\}$). Such spaces model finite cartesian products X^n but are easier to deal with in formalized way (than the said products). This section discusses natural topology defined on $n \to X$ where X is a topological space.

When the index set is finite, the definition of ProductTopBase(I,T) can be simplifed.

lemma fin˙prod˙def˙nat: assumes A1: n∈nat and A2: T –is a topology″
 shows ProductTopBase(n,T) = –FinProd(n→⋃T,W). W∈n→T″
⟨*proof*⟩

A technical lemma providing a formula for finite product on one topological space.

lemma single˙top˙prod: assumes A1: W:1→τ
 shows FinProd(1→⋃τ,W) = ––⟨0,y⟩″. y ∈ W(0)″
⟨*proof*⟩

Intuitively, the topological space of singleton lists valued in X is the same as X. However, each element of this space is a list of length one, i.e a set consisting of a pair $\langle 0, x\rangle$ where x is an element of X. The next lemma provides a formula for the product topology in the corner case when we have only one factor and shows that the product topology of one space is essentially the same as the space.

lemma singleton˙prod˙top: assumes A1: τ –is a topology″
 shows
  SeqProductTopology(1,τ) = –––⟨0,y⟩″. y∈U ″. U∈τ″ and
  IsAhomeomorphism(τ,SeqProductTopology(1,τ),–⟨y,–⟨0,y⟩″⟩.y ∈ ⋃τ″)
⟨*proof*⟩

A special corner case of finite˙top˙prod˙homeo: a space X is homeomorphic to the space of one element lists of X.

theorem singleton˙prod˙top1: assumes A1: τ –is a topology″
 shows IsAhomeomorphism(SeqProductTopology(1,τ),τ,–⟨x,x(0)⟩. x∈1→⋃τ″)
⟨*proof*⟩

A technical lemma describing the carrier of a (cartesian) product topology of the (sequence) product topology of n copies of topology τ and another copy of τ.

lemma finite˙prod˙top: assumes τ –is a topology″ and T = SeqProductTopology(n,τ)
 shows (⋃ProductTopology(T,τ)) = (n→⋃τ)×⋃τ
 ⟨*proof*⟩

If U is a set from the base of X^n and V is open in X, then $U \times V$ is in the base of X^{n+1}. The next lemma is an analogue of this fact for the function space approach.

lemma finite˙prod˙succ˙base: assumes A1: τ –is a topology″ and A2: n ∈ nat and
 A3: U ∈ ProductTopBase(n,τ) and A4: V∈τ

shows {x ∈ succ(n)→$\bigcup\tau$. Init(x) ∈ U ∧ x(n) ∈ V} ∈ ProductTopBase(succ(n),τ)
⟨*proof*⟩

If U is open in X^n and V is open in X, then $U \times V$ is open in X^{n+1}. The next lemma is an analogue of this fact for the function space approach.

lemma finite_prod_succ: assumes A1: τ {is a topology} and A2: n ∈ nat and
A3: U ∈ SeqProductTopology(n,τ) and A4: V∈τ
shows {x ∈ succ(n)→$\bigcup\tau$. Init(x) ∈ U ∧ x(n) ∈ V} ∈ SeqProductTopology(succ(n),τ)
⟨*proof*⟩

In the Topology_ZF_2 theory we define product topology of two topological spaces. The next lemma explains in what sense the topology on finite lists of length n of elements of topological space X can be thought as a model of the product topology on the cartesian product of n copies of that space. Namely, we show that the space of lists of length $n+1$ of elements of X is homeomorphic to the product topology (as defined in Topology_ZF_2) of two spaces: the space of lists of length n and X. Recall that if $\mathcal{B}$ is a base (i.e. satisfies the base condition), then the collection $\{\bigcup B | B \in Pow(\mathcal{B})\}$ is a topology (generated by $\mathcal{B}$).

theorem finite_top_prod_homeo: assumes A1: τ {is a topology} and A2: n ∈ nat and

A3: f = {⟨x,⟨Init(x),x(n)⟩⟩. x ∈ succ(n)→$\bigcup\tau$} and
A4: T = SeqProductTopology(n,τ) and
A5: S = SeqProductTopology(succ(n),τ)
shows IsAhomeomorphism(S,ProductTopology(T,τ),f)
⟨*proof*⟩

end

55 Topology 4

theory Topology_ZF_4 imports Topology_ZF_1 Order_ZF func1 NatOrder_ZF
begin

This theory deals with convergence in topological spaces. Contributed by Daniel de la Concepcion.

55.1 Nets

Nets are a generalization of sequences. It is known that sequences do not determine the behavior of the topological spaces that are not first countable; i.e., have a countable neighborhood base for each point. To solve this problem, nets were defined so that the behavior of any topological space can be thought in terms of convergence of nets.

First we need to define what a directed set is:

definition
IsDirectedSet (˙ directs ˙ 90)
where r directs D ≡ refl(D,r) ∧ trans(r) ∧ (∀ x∈D. ∀ y∈D. ∃ z∈D. ⟨x,z⟩∈r ∧ ⟨y,z⟩∈r)

Any linear order is a directed set; in particular $(\mathbb{N}, \leq)$.

lemma linorder˙imp˙directed:
assumes IsLinOrder(X,r)
shows r directs X
⟨*proof*⟩

Natural numbers are a directed set.

corollary Le˙directs˙nat:
shows IsLinOrder(nat,Le) Le directs nat
⟨*proof*⟩

We are able to define the concept of net, now that we now what a directed set is.

definition
IsNet (˙ –is a net on″ ˙ 90)
where N –is a net on″ X ≡ fst(N):domain(fst(N))→X ∧ (snd(N) directs domain(fst(N))) ∧ domain(fst(N))≠0

Provided a topology and a net directed on its underlying set, we can talk about convergence of the net in the topology.

definition (in topology0)
NetConverges (˙ $\rightarrow_N$ ˙ 90)
where N –is a net on″ ⋃T ⟹ N $\rightarrow_N$ x ≡
(x∈⋃T) ∧ (∀ U∈Pow(⋃T). (x∈int(U) ⟶ (∃ t∈domain(fst(N)). ∀ m∈domain(fst(N)).

(⟨t,m⟩∈snd(N) ⟶ fst(N)m∈U))))

One of the most important directed sets, is the neighborhoods of a point.

theorem (in topology0) directedset˙neighborhoods:
assumes x∈⋃T
defines Neigh≡–U∈Pow(⋃T). x∈int(U)″
defines r≡–⟨U,V⟩∈(Neigh × Neigh). V⊆U″
shows r directs Neigh
⟨*proof*⟩

There can be nets directed by the neighborhoods that converge to the point; if there is a choice function.

theorem (in topology0) net˙direct˙neigh˙converg:
assumes x∈⋃T
defines Neigh≡–U∈Pow(⋃T). x∈int(U)″
defines r≡–⟨U,V⟩∈(Neigh × Neigh). V⊆U″
assumes f:Neigh→⋃T ∀ U∈Neigh. f(U) ∈ U
shows ⟨f,r⟩ $\rightarrow_N$ x
⟨*proof*⟩

55.2 Filters

Nets are a generalization of sequences that can make us see that not all topological spaces can be described by sequences. Nevertheless, nets are not always the tool used to deal with convergence. The reason is that they make use of directed sets which are completely unrelated with the topology.

The topological tools to deal with convergence are what is called filters.

definition
  IsFilter (˙ –is a filter on˝ ˙ 90)
  where $\mathfrak{F}$ –is a filter on˝ X ≡ (0∉$\mathfrak{F}$) ∧ (X∈$\mathfrak{F}$) ∧ ($\mathfrak{F}$⊆Pow(X)) ∧
  (∀A∈$\mathfrak{F}$. ∀B∈$\mathfrak{F}$. A∩B∈$\mathfrak{F}$) ∧ (∀B∈$\mathfrak{F}$. ∀C∈Pow(X). B⊆C ⟶ C∈$\mathfrak{F}$)

Not all the sets of a filter are needed to be consider at all times; as it happens with a topology we can consider bases.

definition
  IsBaseFilter (˙ –is a base filter˝ ˙ 90)
  where C –is a base filter˝ $\mathfrak{F}$ ≡ C⊆$\mathfrak{F}$ ∧ $\mathfrak{F}$=–A∈Pow($\bigcup\mathfrak{F}$). (∃D∈C. D⊆A)˝

Not every set is a base for a filter, as it happens with topologies, there is a condition to be satisfied.

definition
  SatisfiesFilterBase (˙ –satisfies the filter base condition˝ 90)
  where C –satisfies the filter base condition˝ ≡ (∀A∈C. ∀B∈C. ∃D∈C. D⊆A∩B)
∧ C≠0 ∧ 0∉C

Every set of a filter contains a set from the filter's base.

lemma basic˙element˙filter:
  assumes A∈$\mathfrak{F}$ and C –is a base filter˝ $\mathfrak{F}$
  shows ∃D∈C. D⊆A
⟨*proof*⟩

The following two results state that the filter base condition is necessary and sufficient for the filter generated by a base, to be an actual filter. The third result, rewrites the previous two.

theorem basic˙filter˙1:
  assumes C –is a base filter˝ $\mathfrak{F}$ and C –satisfies the filter base condition˝
  shows $\mathfrak{F}$ –is a filter on˝ $\bigcup\mathfrak{F}$
⟨*proof*⟩

A base filter satisfies the filter base condition.

theorem basic˙filter˙2:
  assumes C –is a base filter˝ $\mathfrak{F}$ and $\mathfrak{F}$ –is a filter on˝ $\bigcup\mathfrak{F}$
  shows C –satisfies the filter base condition˝
⟨*proof*⟩

A base filter for a collection satisfies the filter base condition iff that collection is in fact a filter.

theorem basic˙filter:
  assumes C –is a base filter˝ $\mathfrak{F}$
  shows (C –satisfies the filter base condition˝) ⟷ ($\mathfrak{F}$ –is a filter on˝ $\bigcup\mathfrak{F}$)
⟨*proof*⟩

A base for a filter determines a filter up to the underlying set.

theorem base˙unique˙filter:
  assumes C –is a base filter˝ $\mathfrak{F}$1and C –is a base filter˝ $\mathfrak{F}$2
  shows $\mathfrak{F}$1=$\mathfrak{F}$2 ⟷ $\bigcup\mathfrak{F}$1=$\bigcup\mathfrak{F}$2
⟨*proof*⟩

Suppose that we take any nonempty collection C of subsets of some set X. Then this collection is a base filter for the collection of all supersets (in X) of sets from C.

theorem base˙unique˙filter˙set1:
  assumes C ⊆ Pow(X) and C≠0
  shows C –is a base filter˝ –A∈Pow(X). ∃D∈C. D⊆A˝ and $\bigcup$–A∈Pow(X). ∃D∈C. D⊆A˝=X
⟨*proof*⟩

A collection C that satisfies the filter base condition is a base filter for some other collection $\mathfrak{F}$ iff $\mathfrak{F}$ is the collection of supersets of C.

theorem base˙unique˙filter˙set2:
  assumes C⊆Pow(X) and C –satisfies the filter base condition˝
  shows ((C –is a base filter˝ $\mathfrak{F}$) ∧ $\bigcup\mathfrak{F}$=X) ⟷ $\mathfrak{F}$=–A∈Pow(X). ∃D∈C. D⊆A˝
  ⟨*proof*⟩

A simple corollary from the previous lemma.

corollary base˙unique˙filter˙set3:
  assumes C⊆Pow(X) and C –satisfies the filter base condition˝
  shows C –is a base filter˝ –A∈Pow(X). ∃D∈C. D⊆A˝ and $\bigcup$–A∈Pow(X). ∃D∈C. D⊆A˝ = X
⟨*proof*⟩

The convergence for filters is much easier concept to write. Given a topology and a filter on the same underlying set, we can define convergence as containing all the neighborhoods of the point.

definition (in topology0)
  FilterConverges (˙ $\rightarrow_F$ ˙ 50) where
  $\mathfrak{F}$–is a filter on˝$\bigcup$T ⟹ $\mathfrak{F}\rightarrow_F$x ≡
  x∈$\bigcup$T ∧ (–U∈Pow($\bigcup$T). x∈int(U)˝ ⊆ $\mathfrak{F}$)

The neighborhoods of a point form a filter that converges to that point.

lemma (in topology0) neigh˙filter:
  assumes x∈$\bigcup$T
  defines Neigh≡–U∈Pow($\bigcup$T). x∈int(U)˝
  shows Neigh –is a filter on˝$\bigcup$T and Neigh $\rightarrow_F$ x

⟨proof⟩

Note that with the net we built in a previous result, it wasn't clear that we could construct an actual net that converged to the given point without the axiom of choice. With filters, there is no problem.

Another positive point of filters is due to the existence of filter basis. If we have a basis for a filter, then the filter converges to a point iff every neighborhood of that point contains a basic filter element.

theorem (in topology0) convergence˙filter˙base1:
  assumes 𝔉 –is a filter on" ⋃T and C –is a base filter" 𝔉 and 𝔉 $\rightarrow_F$ x
  shows ∀U∈Pow(⋃T). x∈int(U) ⟶ (∃D∈C. D⊆U) and x∈⋃T
⟨proof⟩

A sufficient condition for a filter to converge to a point.

theorem (in topology0) convergence˙filter˙base2:
  assumes 𝔉 –is a filter on" ⋃T and C –is a base filter" 𝔉
    and ∀U∈Pow(⋃T). x∈int(U) ⟶ (∃D∈C. D⊆U) and x∈⋃T
  shows 𝔉 $\rightarrow_F$ x
⟨proof⟩

A necessary and sufficient condition for a filter to converge to a point.

theorem (in topology0) convergence˙filter˙base˙eq:
  assumes 𝔉 –is a filter on" ⋃T and C –is a base filter" 𝔉
  shows (𝔉 $\rightarrow_F$ x) ⟷ ((∀U∈Pow(⋃T). x∈int(U) ⟶ (∃D∈C. D⊆U)) ∧ x∈⋃T)
⟨proof⟩

55.3 Relation between nets and filters

In this section we show that filters do not generalize nets, but still nets and filter are in w way equivalent as far as convergence is considered.

Let's build now a net from a filter, such that both converge to the same points.

definition
  NetOfFilter (Net(˙) 40) where
  𝔉 –is a filter on" ⋃𝔉 ⟹ Net(𝔉) ≡
  ⟨–⟨A,fst(A)⟩. A∈–⟨x,F⟩∈(⋃𝔉)×𝔉. x∈F""",–⟨A,B⟩∈–⟨x,F⟩∈(⋃𝔉)×𝔉. x∈F"×–⟨x,F⟩∈(⋃𝔉)×𝔉. x∈F". snd(B)⊆snd(A)"⟩

Net of a filter is indeed a net.

theorem net˙of˙filter˙is˙net:
  assumes 𝔉 –is a filter on" X
  shows (Net(𝔉)) –is a net on" X
⟨proof⟩

If a filter converges to some point then its net converges to the same point.

theorem (in topology0) filter˙conver˙net˙of˙filter˙conver:

assumes $\mathfrak{F}$ –is a filter on" $\bigcup$T and $\mathfrak{F} \rightarrow_F$ x
shows (Net($\mathfrak{F}$)) $\rightarrow_N$ x
⟨*proof*⟩

If a net converges to a point, then a filter also converges to a point.

theorem (in topology0) net˙of˙filter˙conver˙filter˙conver:
assumes $\mathfrak{F}$ –is a filter on"$\bigcup$T and (Net($\mathfrak{F}$)) $\rightarrow_N$ x
shows $\mathfrak{F} \rightarrow_F$ x
⟨*proof*⟩

A filter converges to a point if and only if its net converges to the point.

theorem (in topology0) filter˙conver˙iff˙net˙of˙filter˙conver:
assumes $\mathfrak{F}$ –is a filter on"$\bigcup$T
shows ($\mathfrak{F} \rightarrow_F$ x) $\longleftrightarrow$ ((Net($\mathfrak{F}$)) $\rightarrow_N$ x)
⟨*proof*⟩

The previous result states that, when considering convergence, the filters do not generalize nets. When considering a filter, there is always a net that converges to the same points of the original filter.

Now we see that with nets, results come naturally applying the axiom of choice; but with filters, the results come, may be less natural, but with no choice. The reason is that Net($\mathfrak{F}$) is a net that doesn't come into our attention as a first choice; maybe because we restrict ourselves to the anti-symmetry property of orders without realizing that a directed set is not an order.

The following results will state that filters are not just a subclass of nets, but that nets and filters are equivalent on convergence: for every filter there is a net converging to the same points, and also, for every net there is a filter converging to the same points.

definition
FilterOfNet (Filter (˙ .. ˙) 40) where
(N –is a net on" X) $\Longrightarrow$ Filter N..X $\equiv$ –A$\in$Pow(X). $\exists$ D$\in$—fst(N)snd(s). s$\in$–s$\in$domain(fst(N))$\times$domain(fst(N)).
s$\in$snd(N) $\wedge$ fst(s)=t0"". t0$\in$domain(fst(N))". D$\subseteq$A"

Filter of a net is indeed a filter

theorem filter˙of˙net˙is˙filter:
assumes N –is a net on" X
shows (Filter N..X) –is a filter on" X and
—fst(N)snd(s). s$\in$–s$\in$domain(fst(N))$\times$domain(fst(N)). s$\in$snd(N) $\wedge$ fst(s)=t0"".
t0$\in$domain(fst(N))" –is a base filter" (Filter N..X)
⟨*proof*⟩

Convergence of a net implies the convergence of the corresponding filter.

theorem (in topology0) net˙conver˙filter˙of˙net˙conver:
assumes N –is a net on" $\bigcup$T and N $\rightarrow_N$ x
shows (Filter N..($\bigcup$T)) $\rightarrow_F$ x
⟨*proof*⟩

Convergence of a filter corresponding to a net implies convergence of the net.

theorem (in topology0) filter˙of˙net˙conver˙net˙conver:
  assumes N –is a net on″ $\bigcup$T and (Filter N..($\bigcup$T)) $\rightarrow_F$ x
  shows N $\rightarrow_N$ x
⟨*proof*⟩

Filter of net converges to a point x if and only the net converges to x.

theorem (in topology0) filter˙of˙net˙conv˙iff˙net˙conv:
  assumes N –is a net on″ $\bigcup$T
  shows ((Filter N..($\bigcup$T)) $\rightarrow_F$ x) $\longleftrightarrow$ (N $\rightarrow_N$ x)
  ⟨*proof*⟩

We know now that filters and nets are the same thing, when working convergence of topological spaces. Sometimes, the nature of filters makes it easier to generalized them as follows.

Instead of considering all subsets of some set X, we can consider only open sets (we get an open filter) or closed sets (we get a closed filter). There are many more useful examples that characterize topological properties.

This type of generalization cannot be done with nets.

Also a filter can give us a topology in the following way:

theorem top˙of˙filter:
  assumes $\mathfrak{F}$ –is a filter on″ $\bigcup\mathfrak{F}$
  shows ($\mathfrak{F}$ ∪ –0″) –is a topology″
⟨*proof*⟩

We can use topology0 locale with filters.

lemma topology0˙filter:
  assumes $\mathfrak{F}$ –is a filter on″ $\bigcup\mathfrak{F}$
  shows topology0($\mathfrak{F}$ ∪ –0″)
  ⟨*proof*⟩

The next abbreviation introduces notation where we want to specify the space where the filter convergence takes place.

abbreviation FilConvTop(˙ $\rightarrow_F$ ˙ –in″ ˙)
  where $\mathfrak{F}$ $\rightarrow_F$ x –in″ T ≡ topology0.FilterConverges(T,$\mathfrak{F}$,x)

The next abbreviation introduces notation where we want to specify the space where the net convergence takes place.

abbreviation NetConvTop(˙ $\rightarrow_N$ ˙ –in″ ˙)
  where N $\rightarrow_N$ x –in″ T ≡ topology0.NetConverges(T,N,x)

Each point of a the union of a filter is a limit of that filter.

lemma lim˙filter˙top˙of˙filter:
  assumes $\mathfrak{F}$ –is a filter on″ $\bigcup\mathfrak{F}$ and x∈$\bigcup\mathfrak{F}$

shows $\mathfrak{F} \rightarrow_F$ x –in" ($\mathfrak{F}$∪–0")
⟨*proof*⟩

end

56 Topology and neighborhoods

theory Topology˙ZF˙4a imports Topology˙ZF˙4
begin

This theory considers the relations between topology and systems of neighborhood filters.

56.1 Neighborhood systems

The standard way of defining a topological space is by specifying a collection of sets that we consider "open" (see the Topology˙ZF theory). An alternative of this approach is to define a collection of neighborhoods for each point of the space.

We define a neighborhood system as a function that takes each point $x \in X$ and assigns it a collection of subsets of X which is called the neighborhoods of x. The neighborhoods of a point x form a filter that satisfies an additional axiom that for every neighborhood N of x we can find another one U such that N is a neighborhood of every point of U.

definition
IsNeighSystem (˙ –is a neighborhood system on" ˙ 90)
where $\mathcal{M}$ –is a neighborhood system on" X ≡ ($\mathcal{M}$: X→Pow(Pow(X))) ∧
(∀ x∈X. ($\mathcal{M}$(x) –is a filter on" X) ∧ (∀ N∈$\mathcal{M}$(x). x∈N ∧ (∃ U∈$\mathcal{M}$(x).∀ y∈U.(N∈$\mathcal{M}$(y))
)))

A neighborhood system on X consists of collections of subsets of X.

lemma neighborhood˙subset:
assumes $\mathcal{M}$ –is a neighborhood system on" X and x∈X and N∈$\mathcal{M}$(x)
shows N⊆X and x∈N
⟨*proof*⟩

Some sources (like Wikipedia) use a bit different definition of neighborhood systems where the U is required to be contained in N. The next lemma shows that this stronger version can be recovered from our definition.

lemma neigh˙def˙stronger:
assumes $\mathcal{M}$ –is a neighborhood system on" X and x∈X and N∈$\mathcal{M}$(x)
shows ∃ U∈$\mathcal{M}$(x).U⊆N ∧ (∀ y∈U.(N∈$\mathcal{M}$(y)))
⟨*proof*⟩

56.2 Topology from neighborhood systems

Given a neighborhood system $\{\mathcal{M}_x\}_{x\in X}$ we can define a topology on X. Namely, we consider a subset of X open if $U \in \mathcal{M}_x$ for every element x of U.

The collection of sets defined as above is indeed a topology.

theorem topology˙from˙neighs:
 assumes $\mathcal{M}$ –is a neighborhood system on″ X
 defines Tdef: T ≡ –U∈Pow(X). ∀x∈U. U ∈ $\mathcal{M}$(x)″
 shows T –is a topology″ and $\bigcup$T = X
⟨*proof*⟩

Some sources (like Wikipedia) define the open sets generated by a neighborhood system "as those sets containing a neighborhood of each of their points". The next lemma shows that this definition is equivalent to the one we are using.

lemma topology˙from˙neighs1:
 assumes $\mathcal{M}$ –is a neighborhood system on″ X
 shows –U∈Pow(X). ∀x∈U. U ∈ $\mathcal{M}$(x)″ = –U∈Pow(X). ∀x∈U. ∃V ∈ $\mathcal{M}$(x). V⊆U″
⟨*proof*⟩

56.3 Neighborhood system from topology

Once we have a topology T we can define a natural neighborhood system on $X = \bigcup T$. In this section we define such neighborhood system and prove its basic properties.

For a topology T we define a neighborhood system of T as a function that takes an $x \in X = \bigcup T$ and assigns it a collection supersets of open sets containing x. We call that the "neighborhood system of T"

definition
 NeighSystem (–neighborhood system of″ ˙ 91)
 where –neighborhood system of″ T ≡ – ⟨x,–V∈Pow($\bigcup$T).∃U∈T.(x∈U ∧ U⊆V)″⟩. x ∈ $\bigcup$T ″

The next lemma shows that open sets are members of (what we will prove later to be) the natural neighborhood system on $X = \bigcup T$.

lemma open˙are˙neighs:
 assumes U∈T x∈U
 shows x ∈ $\bigcup$T and U ∈ –V∈Pow($\bigcup$T).∃U∈T.(x∈U ∧ U⊆V)″
 ⟨*proof*⟩

Another fact we will need is that for every $x \in X = \bigcup T$ the neighborhoods of x form a filter

lemma neighs˙is˙filter:
 assumes T –is a topology″ and x ∈ $\bigcup$T

defines Mdef: $\mathcal{M}$ ≡ –neighborhood system of″ T
shows $\mathcal{M}$(x) –is a filter on″ (⋃T)
⟨*proof*⟩

The next theorem states that the the natural neighborhood system on $X = \bigcup T$ indeed is a neighborhood system.

theorem neigh˙from˙topology:
assumes T –is a topology″
shows (–neighborhood system of″ T) –is a neighborhood system on″ (⋃T)
⟨*proof*⟩

end

57 Topology - examples

theory Topology˙ZF˙examples imports Topology˙ZF Cardinal˙ZF

begin

This theory deals with some concrete examples of topologies.

57.1 CoCardinal Topology

In this section we define and prove the basic properties of the co-cardinal topology on a set X.

The collection of subsets of a set whose complement is strictly bounded by a cardinal is a topology given some assumptions on the cardinal.

definition
CoCardinal(X,T) ≡ –F∈Pow(X). X-F ≺ T″∪ –0″

For any set and any infinite cardinal we prove that CoCardinal(X,Q) forms a topology. The proof is done with an infinite cardinal, but it is obvious that the set Q can be any set equipollent with an infinite cardinal. It is a topology also if the set where the topology is defined is too small or the cardinal too large; in this case, as it is later proved the topology is a discrete topology. And the last case corresponds with Q=1 which translates in the indiscrete topology.

lemma CoCar˙is˙topology:
assumes InfCard (Q)
shows CoCardinal(X,Q) –is a topology″
⟨*proof*⟩

We can use theorems proven in topology0 context for the co-cardinal topology.

theorem topology0˙CoCardinal:

assumes InfCard(T)
shows topology0(CoCardinal(X,T))
⟨*proof*⟩

It can also be proven that if CoCardinal(X,T) is a topology, X≠0, Card(T) and T≠0; then T is an infinite cardinal, X≺T or T=1. It follows from the fact that the union of two closed sets is closed. Choosing the appropriate cardinals, the cofinite and the cocountable topologies are obtained.

The cofinite topology is a very special topology because it is closely related to the separation axiom T_1. It also appears naturally in algebraic geometry.

definition
Cofinite (CoFinite ˙ 90) where
CoFinite X ≡ CoCardinal(X,nat)

Cocountable topology in fact consists of the empty set and all cocountable subsets of X.

definition
Cocountable (CoCountable ˙ 90) where
CoCountable X ≡ CoCardinal(X,csucc(nat))

57.2 Total set, Closed sets, Interior, Closure and Boundary

There are several assertions that can be done to the CoCardinal(X,T) topology. In each case, we will not assume sufficient conditions for CoCardinal(X,T) to be a topology, but they will be enough to do the calculations in every posible case.

The topology is defined in the set X

lemma union˙cocardinal:
assumes T≠0
shows ⋃ CoCardinal(X,T) = X
⟨*proof*⟩

The closed sets are the small subsets of X and X itself.

lemma closed˙sets˙cocardinal:
assumes T≠0
shows D –is closed in˝ CoCardinal(X,T) ⟷ (D∈Pow(X) ∧ D≺T) ∨ D=X
⟨*proof*⟩

The interior of a set is itself if it is open or 0 if it isn't open.

lemma interior˙set˙cocardinal:
assumes noC: T≠0 and A⊆X
shows Interior(A,CoCardinal(X,T))= (if ((X-A) ≺ T) then A else 0)
⟨*proof*⟩

X is a closed set that contains A. This lemma is necessary because we cannot use the lemmas proven in the topology0 context since T≠0˝ is too weak for CoCardinal(X,T) to be a topology.

lemma X˙closedcov˙cocardinal:
  assumes T≠0 A⊆X
  shows X∈ClosedCovers(A,CoCardinal(X,T)) ⟨*proof*⟩

The closure of a set is itself if it is closed or X if it isn't closed.

lemma closure˙set˙cocardinal:
  assumes T≠0A⊆X
  shows Closure(A,CoCardinal(X,T))=(if (A ≺ T) then A else X)
⟨*proof*⟩

The boundary of a set is empty if A and $X - A$ are closed, X if not A neither $X - A$ are closed and; if only one is closed, then the closed one is its boundary.

lemma boundary˙cocardinal:
  assumes T≠0A⊆X
  shows Boundary(A,CoCardinal(X,T)) = (if A≺ T then (if (X-A)≺ T then 0 else A) else (if (X-A)≺ T then X-A else X))
⟨*proof*⟩

If the set is too small or the cardinal too large, then the topology is just the discrete topology.

lemma discrete˙cocardinal:
  assumes X≺T
  shows CoCardinal(X,T) = Pow(X)
⟨*proof*⟩

If the cardinal is taken as T=1 then the topology is indiscrete.

lemma indiscrete˙cocardinal:
  shows CoCardinal(X,1) = –0,X″
⟨*proof*⟩

The topological subspaces of the CoCardinal(X,T) topology are also CoCardinal topologies.

lemma subspace˙cocardinal:
  shows CoCardinal(X,T) –restricted to″ Y = CoCardinal(Y∩X,T)
⟨*proof*⟩

57.3 Excluded Set Topology

In this section, we consider all the subsets of a set which have empty intersection with a fixed set.

The excluded set topology consists of subsets of X that are disjoint with a fixed set U.

definition ExcludedSet(X,U) ≡ –F∈Pow(X). U ∩ F=0″∪ –X″

For any set; we prove that ExcludedSet(X,Q) forms a topology.

theorem excludedset˙is˙topology:
  shows ExcludedSet(X,Q) –is a topology˝
⟨*proof*⟩

We can use topology0 when discussing excluded set topology.

theorem topology0˙excludedset:
  shows topology0(ExcludedSet(X,T))
  ⟨*proof*⟩

Choosing a singleton set, it is considered a point in excluded topology.

definition
  ExcludedPoint(X,p) ≡ ExcludedSet(X,–p˝)

57.4 Total set, closed sets, interior, closure and boundary

Here we discuss what are closed sets, interior, closure and boundary in excluded set topology.

The topology is defined in the set X

lemma union˙excludedset:
  shows $\bigcup$ExcludedSet(X,T) = X
⟨*proof*⟩

The closed sets are those which contain the set (X ∩ T) and 0.

lemma closed˙sets˙excludedset:
  shows D –is closed in˝ExcludedSet(X,T) ⟷ (D∈Pow(X) ∧ (X ∩ T) ⊆ D) ∨ D=0
⟨*proof*⟩

The interior of a set is itself if it is X or the difference with the set T

lemma interior˙set˙excludedset:
  assumes A⊆X
  shows Interior(A,ExcludedSet(X,T)) = (if A=X then X else A-T)
⟨*proof*⟩

The closure of a set is itself if it is 0 or the union with T.

lemma closure˙set˙excludedset:
  assumes A⊆X
  shows Closure(A,ExcludedSet(X,T))=(if A=0 then 0 else A ∪(X∩ T))
⟨*proof*⟩

The boundary of a set is 0 if A is X or 0, and X∩T in other case.

lemma boundary˙excludedset:
  assumes A⊆X
  shows Boundary(A,ExcludedSet(X,T)) = (if A=0∨A=X then 0 else X∩T)
⟨*proof*⟩

57.5 Special cases and subspaces

This section provides some miscellaneous facts about excluded set topologies.

The excluded set topology is equal in the sets T and X∩T.

lemma smaller˙excludedset:
  shows ExcludedSet(X,T) = ExcludedSet(X,(X∩T))
⟨*proof*⟩

If the set which is excluded is disjoint with X, then the topology is discrete.

lemma empty˙excludedset:
  assumes T∩X=0
  shows ExcludedSet(X,T) = Pow(X)
⟨*proof*⟩

The topological subspaces of the ExcludedSet X T topology are also ExcludedSet topologies.

lemma subspace˙excludedset:
  shows ExcludedSet(X,T) –restricted to˝ Y = ExcludedSet(Y ∩ X, T)
⟨*proof*⟩

57.6 Included Set Topology

In this section we consider the subsets of a set which contain a fixed set. The family defined in this section and the one in the previous section are dual; meaning that the closed set of one are the open sets of the other.

We define the included set topology as the collection of supersets of some fixed subset of the space X.

definition
  IncludedSet(X,U) ≡ –F∈Pow(X). U ⊆ F˝ ∪ –0˝

In the next theorem we prove that IncludedSet X Q forms a topology.

theorem includedset˙is˙topology:
  shows IncludedSet(X,Q) –is a topology˝
⟨*proof*⟩

We can reference the theorems proven in the topology0 context when discussing the included set topology.

theorem topology0˙includedset:
  shows topology0(IncludedSet(X,T))
  ⟨*proof*⟩

Choosing a singleton set, it is considered a point excluded topology. In the following lemmas and theorems, when neccessary it will be considered that T≠0 and T⊆X. These cases will appear in the special cases section.

definition
IncludedPoint (IncludedPoint ˙ ˙ 90) where
IncludedPoint X p ≡ IncludedSet(X,–p˝)

57.7 Basic topological notions in included set topology

This section discusses total set, closed sets, interior, closure and boundary for included set topology.

The topology is defined in the set X.

lemma union˙includedset:
assumes T⊆X
shows ⋃IncludedSet(X,T) = X
⟨*proof*⟩

The closed sets are those which are disjoint with T and X.

lemma closed˙sets˙includedset:
assumes T⊆X
shows D –is closed in˝ IncludedSet(X,T) ⟷ (D∈Pow(X) ∧ (D ∩ T)=0)∨ D=X
⟨*proof*⟩

The interior of a set is itself if it is open or the empty set if it isn't.

lemma interior˙set˙includedset:
assumes A⊆X
shows Interior(A,IncludedSet(X,T))= (if T⊆A then A else 0)
⟨*proof*⟩

The closure of a set is itself if it is closed or the whole space if it is not.

lemma closure˙set˙includedset:
assumes A⊆X T⊆X
shows Closure(A,IncludedSet(X,T)) = (if T∩A=0 then A else X)
⟨*proof*⟩

The boundary of a set is X-A if A contains T completely, is A if $X - A$ contains T completely and X if T is divided between the two sets. The case where T=0 is considered as a special case.

lemma boundary˙includedset:
assumes A⊆X T⊆X T≠0
shows Boundary(A,IncludedSet(X,T))=(if T⊆A then X-A else (if T∩A=0 then A else X))
⟨*proof*⟩

57.8 Special cases and subspaces

In this section we discuss some corner cases when some parameters in our definitions are empty and provide some facts about subspaces in included set topologies.

The topology is discrete if T=0

```
lemma smaller˙includedset:
  shows IncludedSet(X,0) = Pow(X)
⟨proof⟩
```

If the set which is included is not a subset of X, then the topology is trivial.

```
lemma empty˙includedset:
  assumes ˜(T⊆X)
  shows IncludedSet(X,T) = –0˝
⟨proof⟩
```

The topological subspaces of the IncludedSet(X,T) topology are also IncludedSet topologies. The trivial case does not fit the idea in the demonstration because if Y⊆X then IncludedSet(Y∩X, Y∩T) is never trivial. There is no need for a separate proof because the only subspace of the trivial topology is itself.

```
lemma subspace˙includedset:
  assumes T⊆X
  shows IncludedSet(X,T) –restricted to˝ Y = IncludedSet(Y∩X,Y∩T)
⟨proof⟩

end
```

58 More examples in topology

```
theory Topology˙ZF˙examples˙1
imports Topology˙ZF˙1 Order˙ZF
begin
```

In this theory file we reformulate the concepts related to a topology in relation with a base of the topology and we give examples of topologies defined by bases or subbases.

58.1 New ideas using a base for a topology

58.2 The topology of a base

Given a family of subsets satisfying the base condition, it is possible to construct a topology where that family is a base of. Even more, it is the only topology with such characteristics.

```
definition
  TopologyWithBase (TopologyBase ˙ 50) where
  U –satisfies the base condition˝ ⟹ TopologyBase U ≡ THE T. U –is a base for˝
T
```

If a collection U of sets satisfies the base condition then the topology constructed from it is indeed a topology and U is a base for this topology.

theorem Base˙topology˙is˙a˙topology:
  assumes U –satisfies the base condition″
  shows (TopologyBase U) –is a topology″ and U –is a base for″ (TopologyBase U)
⟨*proof*⟩

A base doesn't need the empty set.

lemma base˙no˙0:
  shows B–is a base for″T ⟷ (B-–0″)–is a base for″T
⟨*proof*⟩

The interior of a set is the union of all the sets of the base which are fully contained by it.

lemma interior˙set˙base˙topology:
  assumes U –is a base for″ T T–is a topology″
  shows Interior(A,T) = ⋃–T∈U. T⊆A″
⟨*proof*⟩

In the following, we offer another lemma about the closure of a set given a basis for a topology. This lemma is based on cl˙inter˙neigh and inter˙neigh˙cl. It states that it is only necessary to check the sets of the base, not all the open sets.

lemma closure˙set˙base˙topology:
  assumes U –is a base for″ Q Q–is a topology″ A⊆⋃Q
  shows Closure(A,Q) = –x∈⋃Q. ∀T∈U. x∈T⟶A∩T≠0″
⟨*proof*⟩

The restriction of a base is a base for the restriction.

lemma subspace˙base˙topology:
  assumes B –is a base for″ T
  shows (B –restricted to″ Y) –is a base for″ (T –restricted to″ Y)
⟨*proof*⟩

If the base of a topology is contained in the base of another topology, then the topologies maintain the same relation.

theorem base˙subset:
  assumes B–is a base for″TB2–is a base for″T2B⊆B2
  shows T⊆T2
⟨*proof*⟩

58.3 Dual Base for Closed Sets

A dual base for closed sets is the collection of complements of sets of a base for the topology.

definition
  DualBase (DualBase ˙ ˙ 80) where
  B–is a base for″T ⟹ DualBase B T≡–⋃T-U. U∈B″∪–⋃T″

lemma closed˙inter˙dual˙base:
  assumes D–is closed in˝TB–is a base for˝T
  obtains M where M⊆DualBase B TD=⋂M
⟨*proof*⟩

We have already seen for a base that whenever there is a union of open sets, we can consider only basic open sets due to the fact that any open set is a union of basic open sets. What we should expect now is that when there is an intersection of closed sets, we can consider only dual basic closed sets.

lemma closure˙dual˙base:
  assumes U –is a base for˝ QQ–is a topology˝A⊆⋃Q
  shows Closure(A,Q)=⋂–T∈DualBase U Q. A⊆T˝
⟨*proof*⟩

58.4 Partition topology

In the theory file Partitions_ZF.thy; there is a definition to work with partitions. In this setting is much easier to work with a family of subsets.

definition
  IsAPartition (˙–is a partition of˝˙ 90) where
  (U –is a partition of˝ X) ≡ (⋃U=X ∧ (∀A∈U. ∀B∈U. A=B∨ A∩B=0)∧ 0∉U)

A subcollection of a partition is a partition of its union.

lemma subpartition:
  assumes U –is a partition of˝ X V⊆U
  shows V–is a partition of˝⋃V
  ⟨*proof*⟩

A restriction of a partition is a partition. If the empty set appears it has to be removed.

lemma restriction˙partition:
  assumes U –is a partition of˝X
  shows ((U –restricted to˝ Y)-–0˝) –is a partition of˝ (X∩Y)
  ⟨*proof*⟩

Given a partition, the complement of a union of a subfamily is a union of a subfamily.

lemma diff˙union˙is˙union˙diff:
  assumes R⊆P P –is a partition of˝ X
  shows X - ⋃R=⋃(P-R)
⟨*proof*⟩

58.5 Partition topology is a topology.

A partition satisfies the base condition.

lemma partition˙base˙condition:
  assumes P –is a partition of˝ X
  shows P –satisfies the base condition˝
⟨proof⟩

Since a partition is a base of a topology, and this topology is uniquely determined; we can built it. In the definition we have to make sure that we have a partition.

definition
  PartitionTopology (PTopology ˙ ˙ 50) where
  (U –is a partition of˝ X) ⟹ PTopology X U ≡ TopologyBase U

theorem Ptopology˙is˙a˙topology:
  assumes U –is a partition of˝ X
  shows (PTopology X U) –is a topology˝ and U –is a base for˝ (PTopology X U)
  ⟨proof⟩

lemma topology0˙ptopology:
  assumes U –is a partition of˝ X
  shows topology0(PTopology X U)
  ⟨proof⟩

58.6 Total set, Closed sets, Interior, Closure and Boundary

The topology is defined in the set X

lemma union˙ptopology:
  assumes U –is a partition of˝ X
  shows ⋃(PTopology X U)=X
  ⟨proof⟩

The closed sets are the open sets.

lemma closed˙sets˙ptopology:
  assumes T –is a partition of˝ X
  showsD –is closed in˝ (PTopology X T) ⟷ D∈(PTopology X T)
⟨proof⟩

There is a formula for the interior given by an intersection of sets of the dual base. Is the intersection of all the closed sets of the dual basis such that they do not complement A to X. Since the interior of X must be inside X, we have to enter X as one of the sets to be intersected.

lemma interior˙set˙ptopology:
  assumes U –is a partition of˝ XA⊆X
  shows Interior(A,(PTopology X U))=⋂–T∈DualBase U (PTopology X U). T=X∨T∪A≠X˝
⟨proof⟩

The closure of a set is the union of all the sets of the partition which intersect with A.

lemma closure˙set˙ptopology:
 assumes U –is a partition of˝ XA⊆X
 shows Closure(A,(PTopology X U))=⋃–T∈U. T∩A≠0˝
⟨*proof*⟩

The boundary of a set is given by the union of the sets of the partition which have non empty intersection with the set but that are not fully contained in it. Another equivalent statement would be: the union of the sets of the partition which have non empty intersection with the set and its complement.

lemma boundary˙set˙ptopology:
 assumes U –is a partition of˝ XA⊆X
 shows Boundary(A,(PTopology X U))=⋃–T∈U. T∩A≠0 ∧ ˜(T⊆A)˝
⟨*proof*⟩

58.7 Special cases and subspaces

The discrete and the indiscrete topologies appear as special cases of this partition topologies.

lemma discrete˙partition:
 shows —x˝.x∈X˝ –is a partition of˝X
 ⟨*proof*⟩

lemma indiscrete˙partition:
 assumes X≠0
 shows –X˝ –is a partition of˝ X
 ⟨*proof*⟩

theorem discrete˙ptopology:
 shows (PTopology X —x˝.x∈X˝)=Pow(X)
⟨*proof*⟩

theorem indiscrete˙ptopology:
 assumes X≠0
 shows (PTopology X –X˝)=–0,X˝
⟨*proof*⟩

The topological subspaces of the (PTopology X U) are partition topologies.

lemma subspace˙ptopology:
 assumes U–is a partition of˝X
 shows (PTopology X U) –restricted to˝ Y=(PTopology (X∩Y) ((U –restricted to˝ Y)-–0˝))
⟨*proof*⟩

58.8 Order topologies

58.9 Order topology is a topology

Given a totally ordered set, several topologies can be defined using the order relation. First we define an open interval, notice that the set defined as Interval is a closed interval; and open rays.

definition
  IntervalX where
  IntervalX(X,r,b,c)≡(Interval(r,b,c)∩X)-–b,c″
definition
  LeftRayX where
  LeftRayX(X,r,b)≡–c∈X. ⟨c,b⟩∈r″-–b″
definition
  RightRayX where
  RightRayX(X,r,b)≡–c∈X. ⟨b,c⟩∈r″-–b″

Intersections of intervals and rays.

lemma inter˙two˙intervals:
  assumes bu∈Xbv∈Xcu∈Xcv∈XIsLinOrder(X,r)
  shows IntervalX(X,r,bu,cu)∩IntervalX(X,r,bv,cv)=IntervalX(X,r,GreaterOf(r,bu,bv),SmallerOf(r,cu,cv))
⟨*proof*⟩

lemma inter˙rray˙interval:
  assumes bv∈Xbu∈Xcv∈XIsLinOrder(X,r)
  shows RightRayX(X,r,bu)∩IntervalX(X,r,bv,cv)=IntervalX(X,r,GreaterOf(r,bu,bv),cv)
⟨*proof*⟩

lemma inter˙lray˙interval:
  assumes bv∈Xcu∈Xcv∈XIsLinOrder(X,r)
  shows LeftRayX(X,r,cu)∩IntervalX(X,r,bv,cv)=IntervalX(X,r,bv,SmallerOf(r,cu,cv))
⟨*proof*⟩

lemma inter˙lray˙rray:
  assumes bu∈Xcv∈XIsLinOrder(X,r)
  shows LeftRayX(X,r,bu)∩RightRayX(X,r,cv)=IntervalX(X,r,cv,bu)
  ⟨*proof*⟩

lemma inter˙lray˙lray:
  assumes bu∈Xcv∈XIsLinOrder(X,r)
  shows LeftRayX(X,r,bu)∩LeftRayX(X,r,cv)=LeftRayX(X,r,SmallerOf(r,bu,cv))
⟨*proof*⟩

lemma inter˙rray˙rray:
  assumes bu∈Xcv∈XIsLinOrder(X,r)
  shows RightRayX(X,r,bu)∩RightRayX(X,r,cv)=RightRayX(X,r,GreaterOf(r,bu,cv))
⟨*proof*⟩

The open intervals and rays satisfy the base condition.

lemma intervals˙rays˙base˙condition:
  assumes IsLinOrder(X,r)
  shows –IntervalX(X,r,b,c). ⟨b,c⟩∈X×X″∪–LeftRayX(X,r,b). b∈X″∪–RightRayX(X,r,b). b∈X″ –satisfies the base condition″
⟨*proof*⟩

Since the intervals and rays form a base of a topology, and this topology is uniquely determined; we can built it. In the definition we have to make sure that we have a totally ordered set.

definition
  OrderTopology (OrdTopology ˙ ˙ 50) where
  IsLinOrder(X,r) ⟹ OrdTopology X r ≡ TopologyBase –IntervalX(X,r,b,c). ⟨b,c⟩∈X×X″∪–LeftRayX(X,r,b). b∈X″∪–RightRayX(X,r,b). b∈X″

theorem Ordtopology˙is˙a˙topology:
  assumes IsLinOrder(X,r)
  shows (OrdTopology X r) –is a topology″ and –IntervalX(X,r,b,c). ⟨b,c⟩∈X×X″∪–LeftRayX(X,r,b). b∈X″∪–RightRayX(X,r,b). b∈X″ –is a base for″ (OrdTopology X r)
  ⟨*proof*⟩

lemma topology0˙ordtopology:
  assumes IsLinOrder(X,r)
  shows topology0(OrdTopology X r)
  ⟨*proof*⟩

58.10 Total set

The topology is defined in the set X, when X has more than one point

lemma union˙ordtopology:
  assumes IsLinOrder(X,r)∃x y. x≠y ∧ x∈X∧ y∈X
  shows ⋃(OrdTopology X r)=X
⟨*proof*⟩

The interior, closure and boundary can be calculated using the formulas proved in the section that deals with the base.

The subspace of an order topology doesn't have to be an order topology.

58.11 Right order and Left order topologies.

Notice that the left and right rays are closed under intersection, hence they form a base of a topology. They are called right order topology and left order topology respectively.

If the order in X has a minimal or a maximal element, is necessary to consider X as an element of the base or that limit point wouldn't be in any basic open set.

58.11.1 Right and Left Order topologies are topologies

lemma leftrays˙base˙condition:
assumes IsLinOrder(X,r)
shows –LeftRayX(X,r,b). b∈X″∪–X″ –satisfies the base condition″
⟨*proof*⟩

lemma rightrays˙base˙condition:
assumes IsLinOrder(X,r)
shows –RightRayX(X,r,b). b∈X″∪–X″ –satisfies the base condition″
⟨*proof*⟩

definition
 LeftOrderTopology (LOrdTopology ˙ ˙ 50) where
 IsLinOrder(X,r) ⟹ LOrdTopology X r ≡ TopologyBase –LeftRayX(X,r,b). b∈X″∪–X″

definition
 RightOrderTopology (ROrdTopology ˙ ˙ 50) where
 IsLinOrder(X,r) ⟹ ROrdTopology X r ≡ TopologyBase –RightRayX(X,r,b). b∈X″∪–X″

theorem LOrdtopology˙ROrdtopology˙are˙topologies:
 assumes IsLinOrder(X,r)
 shows (LOrdTopology X r) –is a topology″ and –LeftRayX(X,r,b). b∈X″∪–X″ –is a base for″ (LOrdTopology X r)
 and (ROrdTopology X r) –is a topology″ and –RightRayX(X,r,b). b∈X″∪–X″ –is a base for″ (ROrdTopology X r)
 ⟨*proof*⟩

lemma topology0˙lordtopology˙rordtopology:
 assumes IsLinOrder(X,r)
 shows topology0(LOrdTopology X r) and topology0(ROrdTopology X r)
 ⟨*proof*⟩

58.11.2 Total set

The topology is defined on the set X

lemma union˙lordtopology˙rordtopology:
 assumes IsLinOrder(X,r)
 shows ⋃(LOrdTopology X r)=X and ⋃(ROrdTopology X r)=X
 ⟨*proof*⟩

58.12 Union of Topologies

The union of two topologies is not a topology. A way to overcome this fact is to define the following topology:

definition
 joinT (joinT ˙ 90) where

(∀T∈M. T–is a topology″ ∧ (∀Q∈M. ⋃Q=⋃T)) ⟹ (joinT M ≡ THE T. (⋃M)–is a subbase for″ T)

First let's proof that given a family of sets, then it is a subbase for a topology.

The first result states that from any family of sets we get a base using finite intersections of them. The second one states that any family of sets is a subbase of some topology.

theorem subset˙as˙subbase:
  shows –⋂A. A ∈ FinPow(B)″ –satisfies the base condition″
⟨*proof*⟩

theorem Top˙subbase:
  assumes T = –⋃A. A∈Pow(–⋂A. A ∈ FinPow(B)″)″
  shows T –is a topology″ and B –is a subbase for″ T
⟨*proof*⟩

A subbase defines a unique topology.

theorem same˙subbase˙same˙top:
  assumes B –is a subbase for″ T and B –is a subbase for″ S
  shows T = S
  ⟨*proof*⟩

end

59 Properties in Topology

theory Topology˙ZF˙properties imports Topology˙ZF˙examples Topology˙ZF˙examples˙1

begin

This theory deals with topological properties which make use of cardinals.

59.1 Properties of compactness

It is already defined what is a compact topological space, but the is a generalization which may be useful sometimes.

definition
  IsCompactOfCard (˙–is compact of cardinal″˙ –in″˙ 90)
  where K–is compact of cardinal″ Q–in″T ≡ (Card(Q) ∧ K ⊆ ⋃T ∧
  (∀ M∈Pow(T). K ⊆ ⋃M ⟶ (∃ N ∈ Pow(M). K ⊆ ⋃N ∧ N≺Q)))

The usual compact property is the one defined over the cardinal of the natural numbers.

lemma Compact˙is˙card˙nat:
  shows K–is compact in″T ⟷ (K–is compact of cardinal″ nat –in″T)
⟨*proof*⟩

Another property of this kind widely used is the Lindeloef property; it is the one on the successor of the natural numbers.

definition
  IsLindeloef (˙–is lindeloef in˝˙ 90) where
  K –is lindeloef in˝ T≡K–is compact of cardinal˝csucc(nat)–in˝T

It would be natural to think that every countable set with any topology is Lindeloef; but this statement is not provable in ZF. The reason is that to build a subcover, most of the time we need to choose sets from an infinite collection which cannot be done in ZF. Additional axioms are needed, but strictly weaker than the axiom of choice.

However, if the topology has not many open sets, then the topological space is indeed compact.

theorem card˙top˙comp:
  assumes Card(Q) T≺Q K⊆⋃T
  shows (K)–is compact of cardinal˝Q–in˝T
⟨*proof*⟩

The union of two compact sets, is compact; of any cardinality.

theorem union˙compact:
  assumes K–is compact of cardinal˝Q–in˝T K1–is compact of cardinal˝Q–in˝T InfCard(Q)
  shows (K ∪ K1)–is compact of cardinal˝Q–in˝T ⟨*proof*⟩

If a set is compact of cardinality Q for some topology, it is compact of cardinality Q for every coarser topology.

theorem compact˙coarser:
  assumes T1⊆T and ⋃T1=⋃T and (K)–is compact of cardinal˝Q–in˝T
  shows (K)–is compact of cardinal˝Q–in˝T1
⟨*proof*⟩

If some set is compact for some cardinal, it is compact for any greater cardinal.

theorem compact˙greater˙card:
  assumes Q≲Q1 and (K)–is compact of cardinal˝Q–in˝T and Card(Q1)
  shows (K)–is compact of cardinal˝Q1–in˝T
⟨*proof*⟩

A closed subspace of a compact space of any cardinality, is also compact of the same cardinality.

theorem compact˙closed:
  assumes K –is compact of cardinal˝ Q –in˝ T
    and R –is closed in˝ T
  shows (K∩R) –is compact of cardinal˝ Q –in˝ T
⟨*proof*⟩

59.2 Properties of numerability

The properties of numerability deal with cardinals of some sets built from the topology. The properties which are normally used are the ones related to the cardinal of the natural numbers or its successor.

definition
  IsFirstOfCard (˙ –is of first type of cardinal"˙ 90) where
  (T –is of first type of cardinal" Q) ≡ ∀x∈⋃T. (∃B. (B –is a base for" T) ∧ (–b∈B. x∈b" ≺ Q))

definition
  IsSecondOfCard (˙ –is of second type of cardinal"˙ 90) where
  (T –is of second type of cardinal"Q) ≡ (∃B. (B –is a base for" T) ∧ (B ≺ Q))

definition
  IsSeparableOfCard (˙–is separable of cardinal"˙ 90) where
  T–is separable of cardinal"Q≡ ∃U∈Pow(⋃T). Closure(U,T)=⋃T ∧ U≺Q

definition
  IsFirstCountable (˙ –is first countable" 90) where
  (T –is first countable") ≡ T –is of first type of cardinal" csucc(nat)

definition
  IsSecondCountable (˙ –is second countable" 90) where
  (T –is second countable") ≡ (T –is of second type of cardinal"csucc(nat))

definition
  IsSeparable (˙–is separable" 90) where
  T–is separable"≡ T–is separable of cardinal"csucc(nat)

If a set is of second type of cardinal Q, then it is of first type of that same cardinal.

theorem second˙imp˙first:
  assumes T–is of second type of cardinal"Q
  shows T–is of first type of cardinal"Q
⟨*proof*⟩

A set is dense iff it intersects all non-empty, open sets of the topology.

lemma dense˙int˙open:
  assumes T–is a topology" and A⊆⋃T
  shows Closure(A,T)=⋃T ⟷ (∀U∈T. U≠0 ⟶ A∩U≠0)
⟨*proof*⟩

59.3 Relations between numerability properties and choice principles

It is known that some statements in topology aren't just derived from choice axioms, but also equivalent to them. Here is an example

The following are equivalent:

- Every topological space of second cardinality csucc(Q) is separable of cardinality csucc(Q).
- The axiom of Q choice.

In the article [4] there is a proof of this statement for Q= $\mathbb{N}$, with more equivalences.

If a topology is of second type of cardinal csucc(Q), then it is separable of the same cardinal. This result makes use of the axiom of choice for the cardinal Q on subsets of $\bigcup$T.

theorem Q˙choice˙imp˙second˙imp˙separable:
 assumes T–is of second type of cardinal″csucc(Q)
  and –the axiom of″ Q –choice holds for subsets″ $\bigcup$T
  and T–is a topology″
 shows T–is separable of cardinal″csucc(Q)
⟨*proof*⟩

The next theorem resolves that the axiom of Q choice for subsets of $\bigcup$T is necessary for second type spaces to be separable of the same cardinal csucc(Q).

theorem second˙imp˙separable˙imp˙Q˙choice:
 assumes ∀T. (T–is a topology″ ∧ (T–is of second type of cardinal″csucc(Q))) ⟶ (T–is separable of cardinal″csucc(Q))
 and Card(Q)
 shows –the axiom of″ Q –choice holds″
⟨*proof*⟩

Here is the equivalence from the two previous results.

theorem Q˙choice˙eq˙secon˙imp˙sepa:
 assumes Card(Q)
 shows (∀T. (T–is a topology″ ∧ (T–is of second type of cardinal″csucc(Q))) ⟶ (T–is separable of cardinal″csucc(Q)))
  ⟷(–the axiom of″ Q –choice holds″)
 ⟨*proof*⟩

Given a base injective with a set, then we can find a base whose elements are indexed by that set.

lemma base˙to˙indexed˙base:
 assumes B $\lesssim$Q B –is a base for″T
 shows ∃N. –Ni. i∈Q″–is a base for″T
⟨*proof*⟩

59.4 Relation between numerability and compactness

If the axiom of Q choice holds, then any topology of second type of cardinal csucc(Q) is compact of cardinal csucc(Q)

theorem compact˙of˙cardinal˙Q:
  assumes –the axiom of″ Q –choice holds for subsets″ (Pow(Q))
    T–is of second type of cardinal″csucc(Q)
    T–is a topology″
  shows (($\bigcup$T)–is compact of cardinal″csucc(Q)–in″T)
⟨*proof*⟩

In the following proof, we have chosen an infinite cardinal to be able to apply the equation Q $\times$ Q $\approx$ Q. For finite cardinals; both, the assumption and the axiom of choice, are always true.

theorem second˙imp˙compact˙imp˙Q˙choice˙PowQ:
  assumes $\forall$T. (T–is a topology″ $\wedge$ (T–is of second type of cardinal″csucc(Q))) $\longrightarrow$ (($\bigcup$T)–is compact of cardinal″csucc(Q)–in″T)
  and InfCard(Q)
  shows –the axiom of″ Q –choice holds for subsets″ (Pow(Q))
⟨*proof*⟩

The two previous results, state the following equivalence:

theorem Q˙choice˙Pow˙eq˙secon˙imp˙comp:
  assumes InfCard(Q)
  shows ($\forall$T. (T–is a topology″ $\wedge$ (T–is of second type of cardinal″csucc(Q))) $\longrightarrow$ (($\bigcup$T)–is compact of cardinal″csucc(Q)–in″T))
    $\longleftrightarrow$(–the axiom of″ Q –choice holds for subsets″ (Pow(Q)))
    ⟨*proof*⟩

In the next result we will prove that if the space $(\kappa, Pow(\kappa))$, for κ an infinite cardinal, is compact of its successor cardinal; then all topologycal spaces which are of second type of the successor cardinal of κ are also compact of that cardinal.

theorem Q˙csuccQ˙comp˙eq˙Q˙choice˙Pow:
  assumes InfCard(Q) (Q)–is compact of cardinal″csucc(Q)–in″Pow(Q)
  shows $\forall$T. (T–is a topology″ $\wedge$ (T–is of second type of cardinal″csucc(Q))) $\longrightarrow$ (($\bigcup$T)–is compact of cardinal″csucc(Q)–in″T)
⟨*proof*⟩

theorem Q˙disc˙is˙second˙card˙csuccQ:
  assumes InfCard(Q)
  shows Pow(Q)–is of second type of cardinal″csucc(Q)
⟨*proof*⟩

This previous results give us another equivalence of the axiom of Q choice that is apparently weaker (easier to check) to the previous one.

theorem Q˙disc˙comp˙csuccQ˙eq˙Q˙choice˙csuccQ:
  assumes InfCard(Q)
  shows (Q–is compact of cardinal″csucc(Q)–in″(Pow(Q))) $\longleftrightarrow$ (–the axiom of″Q–choice holds for subsets″(Pow(Q)))
  ⟨*proof*⟩

end

60 Topology 5

theory Topology˙ZF˙5 imports Topology˙ZF˙properties Topology˙ZF˙examples˙1 Topology˙ZF˙4
begin

60.1 Some results for separation axioms

First we will give a global characterization of T_1-spaces; which is interesting because it involves the cardinal $\mathbb{N}$.

lemma (in topology0) T1˙cocardinal˙coarser:
 shows (T –is T_1″) $\longleftrightarrow$ (CoFinite ($\bigcup$T))$\subseteq$T
⟨*proof*⟩

In the previous proof, it is obvious that we don't need to check if ever cofinite set is open. It is enough to check if every singleton is closed.

corollary(in topology0) T1˙iff˙singleton˙closed:
 shows (T –is T_1″) $\longleftrightarrow$ ($\forall$x$\in\bigcup$T. –x″–is closed in″T)
⟨*proof*⟩

Secondly, let's show that the CoCardinal X Q topologies for different sets Q are all ordered as the partial order of sets. (The order is linear when considering only cardinals)

lemma order˙cocardinal˙top:
 fixes X
 assumes Q1$\lesssim$Q2
 shows CoCardinal(X,Q1) $\subseteq$ CoCardinal(X,Q2)
⟨*proof*⟩

corollary cocardinal˙is˙T1:
 fixes X K
 assumes InfCard(K)
 shows CoCardinal(X,K) –is T_1″
⟨*proof*⟩

In T_2-spaces, filters and nets have at most one limit point.

lemma (in topology0) T2˙imp˙unique˙limit˙filter:
 assumes T –is T_2″ $\mathfrak{F}$ –is a filter on″$\bigcup$T $\mathfrak{F}$ $\rightarrow_F$ x $\mathfrak{F}$ $\rightarrow_F$ y
 shows x=y
⟨*proof*⟩

lemma (in topology0) T2˙imp˙unique˙limit˙net:
 assumes T –is T_2″ N –is a net on″$\bigcup$T N $\rightarrow_N$ x N $\rightarrow_N$ y
 shows x=y

⟨*proof*⟩

In fact, T_2-spaces are characterized by this property. For this proof we build a filter containing the union of two filters.

lemma (in topology0) unique˙limit˙filter˙imp˙T2:
 assumes $\forall x\in\bigcup T. \forall y\in\bigcup T. \forall \mathfrak{F}. ((\mathfrak{F}$ –is a filter on″$\bigcup T) \wedge (\mathfrak{F} \rightarrow_F x) \wedge (\mathfrak{F} \rightarrow_F y)) \longrightarrow x=y$
 shows T –is T_2″
⟨*proof*⟩

lemma (in topology0) unique˙limit˙net˙imp˙T2:
 assumes $\forall x\in\bigcup T. \forall y\in\bigcup T. \forall N. ((N$ –is a net on″$\bigcup T) \wedge (N \rightarrow_N x) \wedge (N \rightarrow_N y)) \longrightarrow x=y$
 shows T –is T_2″
⟨*proof*⟩

This results make easy to check if a space is T_2.

The topology which comes from a filter as in $\mathfrak{F}$ –is a filter on″ $\bigcup\mathfrak{F} \Longrightarrow (\mathfrak{F} \cup$ –0″) –is a topology″ is not T_2 generally. We will see in this file later on, that the exceptions are a consequence of the spectrum.

corollary filter˙T2˙imp˙card1:
 assumes ($\mathfrak{F}\cup$–0″) –is T_2″ $\mathfrak{F}$ –is a filter on″ $\bigcup\mathfrak{F}$ $x\in\bigcup\mathfrak{F}$
 shows $\bigcup\mathfrak{F}$=–x″
⟨*proof*⟩

There are more separation axioms that just T_0, T_1 or T_2

definition
 IsRegular (˙–is regular″ 90)
 where T–is regular″ $\equiv \forall A$. A–is closed in″T $\longrightarrow (\forall x\in\bigcup T\text{-}A. \exists U\in T. \exists V\in T. A\subseteq U\wedge x\in V\wedge U\cap V=0)$

definition
 isT3 (˙–is T_3″ 90)
 where T–is T_3″ $\equiv$ (T–is T_1″) $\wedge$ (T–is regular″)

definition
 IsNormal (˙–is normal″ 90)
 where T–is normal″ $\equiv \forall A$. A–is closed in″T $\longrightarrow (\forall B$. B–is closed in″T $\wedge A\cap B=0 \longrightarrow$
 $(\exists U\in T. \exists V\in T. A\subseteq U\wedge B\subseteq V\wedge U\cap V=0))$

definition
 isT4 (˙–is T_4″ 90)
 where T–is T_4″ $\equiv$ (T–is T_1″) $\wedge$ (T–is normal″)

lemma (in topology0) T4˙is˙T3:
 assumes T–is T_4″ shows T–is T_3″
⟨*proof*⟩

lemma (in topology0) T3_is_T2:
  assumes T{is T_3} shows T{is T_2}
⟨proof⟩

Regularity can be rewritten in terms of existence of certain neighboorhoods.

lemma (in topology0) regular_imp_exist_clos_neig:
  assumes T{is regular} and U∈T and x∈U
  shows ∃V∈T. x∈V ∧ cl(V)⊆U
⟨proof⟩

lemma (in topology0) exist_clos_neig_imp_regular:
  assumes ∀x∈⋃T. ∀U∈T. x∈U ⟶ (∃V∈T. x∈V∧ cl(V)⊆U)
  shows T{is regular}
⟨proof⟩

lemma (in topology0) regular_eq:
  shows T{is regular} ⟷ (∀x∈⋃T. ∀U∈T. x∈U ⟶ (∃V∈T. x∈V∧ cl(V)⊆U))
  ⟨proof⟩

A Hausdorff space separates compact spaces from points.

theorem (in topology0) T2_compact_point:
  assumes T{is T_2} A{is compact in}T x∈⋃T x∉A
  shows ∃U∈T. ∃V∈T. A⊆U ∧ x∈V ∧ U∩V=0
⟨proof⟩

A Hausdorff space separates compact spaces from other compact spaces.

theorem (in topology0) T2_compact_compact:
  assumes T{is T_2} A{is compact in}T B{is compact in}T A∩B=0
  shows ∃U∈T. ∃V∈T. A⊆U ∧ B⊆V ∧ U∩V=0
⟨proof⟩

A compact Hausdorff space is normal.

corollary (in topology0) T2_compact_is_normal:
  assumes T{is T_2} (⋃T){is compact in}T
  shows T{is normal} ⟨proof⟩

60.2 Hereditability

A topological property is hereditary if whenever a space has it, every subspace also has it.

definition IsHer (_{is hereditary} 90)
  where P {is hereditary} ≡ ∀T. T{is a topology} ∧ P(T) ⟶ (∀A∈Pow(⋃T). P(T{restricted to}A))

lemma subspace_of_subspace:
  assumes A⊆BB⊆⋃T
  shows T{restricted to}A=(T{restricted to}B){restricted to}A

⟨proof⟩

The separation properties T_0, T_1, T_2 y T_3 are hereditary.

theorem regular˙here:
 assumes T–is regular″ A∈Pow(⋃T) shows (T–restricted to″A)–is regular″
⟨proof⟩

corollary here˙regular:
 shows IsRegular –is hereditary″ ⟨proof⟩

theorem T1˙here:
 assumes T–is T_1″ A∈Pow(⋃T) shows (T–restricted to″A)–is T_1″
⟨proof⟩

corollary here˙T1:
 shows isT1 –is hereditary″ ⟨proof⟩

lemma here˙and:
 assumes P –is hereditary″ Q –is hereditary″
 shows (λT. P(T) ∧ Q(T)) –is hereditary″ ⟨proof⟩

corollary here˙T3:
 shows isT3 –is hereditary″ ⟨proof⟩

lemma T2˙here:
 assumes T–is T_2″ A∈Pow(⋃T) shows (T–restricted to″A)–is T_2″
⟨proof⟩

corollary here˙T2:
 shows isT2 –is hereditary″ ⟨proof⟩

lemma T0˙here:
 assumes T–is T_0″ A∈Pow(⋃T) shows (T–restricted to″A)–is T_0″
⟨proof⟩

corollary here˙T0:
 shows isT0 –is hereditary″ ⟨proof⟩

60.3 Spectrum and anti-properties

The spectrum of a topological property is a class of sets such that all topologies defined over that set have that property.

The spectrum of a property gives us the list of sets for which the property doesn't give any topological information. Being in the spectrum of a topological property is an invariant in the category of sets and function; mening that equipollent sets are in the same spectra.

definition Spec (˙ –is in the spectrum of″ ˙ 99) where Spec(K,P) ≡ ∀T. ((T–is a topology″ ∧ ⋃T≈K) ⟶ P(T))

lemma equipollent˙spect:
assumes A≈B B –is in the spectrum of″ P
shows A –is in the spectrum of″ P
⟨*proof*⟩

theorem eqpoll˙iff˙spec:
assumes A≈B
shows (B –is in the spectrum of″ P) ⟷ (A –is in the spectrum of″ P)
⟨*proof*⟩

From the previous statement, we see that the spectrum could be formed only by representative of clases of sets. If AC holds, this means that the spectrum can be taken as a set or class of cardinal numbers.

Here is an example of the spectrum. The proof lies in the indiscrite filter –A″ that can be build for any set. In this proof, we see that without choice, there is no way to define the sepctrum of a property with cardinals because if a set is not comparable with any ordinal, its cardinal is defined as 0 without the set being empty.

theorem T4˙spectrum:
shows (A –is in the spectrum of″ isT4) ⟷ A ≲ 1
⟨*proof*⟩

If the topological properties are related, then so are the spectra.

lemma P˙imp˙Q˙spec˙inv:
assumes ∀T. T–is a topology″ ⟶ (Q(T) ⟶ P(T)) A –is in the spectrum of″ Q
shows A –is in the spectrum of″ P
⟨*proof*⟩

Since we already now the spectrum of T_4; if we now the spectrum of T_0, it should be easier to compute the spectrum of T_1, T_2 and T_3.

theorem T0˙spectrum:
shows (A –is in the spectrum of″ isT0) ⟷ A ≲ 1
⟨*proof*⟩

theorem T1˙spectrum:
shows (A –is in the spectrum of″ isT1) ⟷ A ≲ 1
⟨*proof*⟩

theorem T2˙spectrum:
shows (A –is in the spectrum of″ isT2) ⟷ A ≲ 1
⟨*proof*⟩

theorem T3˙spectrum:
shows (A –is in the spectrum of″ isT3) ⟷ A ≲ 1
⟨*proof*⟩

theorem compact˙spectrum:
shows (A –is in the spectrum of" (λT. ($\bigcup$T) –is compact in"T)) $\longleftrightarrow$ Finite(A)
⟨*proof*⟩

It is, at least for some people, surprising that the spectrum of some properties cannot be completely determined in ZF.

theorem compactK˙spectrum:
assumes –the axiom of"K–choice holds for subsets"(Pow(K)) Card(K)
shows (A –is in the spectrum of" (λT. (($\bigcup$T)–is compact of cardinal" csucc(K)–in"T))) $\longleftrightarrow$ (A$\lesssim$K)
⟨*proof*⟩

theorem compactK˙spectrum˙reverse:
assumes $\forall$A. (A –is in the spectrum of" (λT. (($\bigcup$T)–is compact of cardinal" csucc(K)–in"T))) $\longleftrightarrow$ (A$\lesssim$K) InfCard(K)
shows –the axiom of"K–choice holds for subsets"(Pow(K))
⟨*proof*⟩

This last theorem states that if one of the forms of the axiom of choice related to this compactness property fails, then the spectrum will be different. Notice that even for Lindelf spaces that will happend.

The spectrum gives us the posibility to define what an anti-property means. A space is anti-P if the only subspaces which have the property are the ones in the spectrum of P. This concept tries to put together spaces that are completely opposite to spaces where P(T).

definition
antiProperty (˙–is anti-"˙ 50)
where T–is anti-"P $\equiv$ $\forall$A$\in$Pow($\bigcup$T). P(T–restricted to"A) $\longrightarrow$ (A –is in the spectrum of" P)

abbreviation
ANTI(P) $\equiv$ λT. (T–is anti-"P)

A first, very simple, but very useful result is the following: when the properties are related and the spectra are equal, then the anti-properties are related in the oposite direction.

theorem (in topology0) eq˙spect˙rev˙imp˙anti:
assumes $\forall$T. T–is a topology" $\longrightarrow$ P(T) $\longrightarrow$ Q(T) $\forall$A. (A–is in the spectrum of"Q) $\longrightarrow$ (A–is in the spectrum of"P)
and T–is anti-"Q
shows T–is anti-"P
⟨*proof*⟩

If a space can be P(T)$\wedge$Q(T) only in case the underlying set is in the spectrum of P; then Q(T)$\longrightarrow$ANTI(P,T) when Q is hereditary.

theorem Q˙P˙imp˙Spec:

assumes ∀T. ((T–is a topology″∧P(T)∧Q(T))⟶ ((⋃T)–is in the spectrum of″P))
and Q–is hereditary″
shows ∀T. T–is a topology″ ⟶ (Q(T)⟶(T–is anti-″P))
⟨*proof*⟩

If a topologycal space has an hereditary property, then it has its double-anti property.

theorem (in topology0)her˙P˙imp˙anti2P:
assumes P–is hereditary″ P(T)
shows T–is anti-″ANTI(P)
⟨*proof*⟩

The anti-properties are always hereditary

theorem anti˙here:
shows ANTI(P)–is hereditary″
⟨*proof*⟩

corollary (in topology0) anti˙imp˙anti3:
assumes T–is anti-″P
shows T–is anti-″ANTI(ANTI(P))
⟨*proof*⟩

In the article [5], we can find some results on anti-properties.

theorem (in topology0) anti˙T0:
shows (T–is anti-″isT0) ⟷ T=–0,⋃T″
⟨*proof*⟩

lemma indiscrete˙spectrum:
shows (A –is in the spectrum of″(λT. T=–0,⋃T″)) ⟷ A≲1
⟨*proof*⟩

theorem (in topology0) anti˙indiscrete:
shows (T–is anti-″(λT. T=–0,⋃T″)) ⟷ T–is T_0″
⟨*proof*⟩

The conclusion is that being T_0 is just the opposite to being indiscrete.

Next, let's compute the anti-T_i for $i = 1,\ 2,\ 3$ or 4. Surprisingly, they are all the same. Meaning, that the total negation of T_1 is enough to negate all of these axioms.

theorem anti˙T1:
shows (T–is anti-″isT1) ⟷ (IsLinOrder(T,–⟨U,V⟩∈Pow(⋃T)×Pow(⋃T). U⊆V″))
⟨*proof*⟩

corollary linordtop˙here:
shows (λT. IsLinOrder(T,–⟨U,V⟩∈Pow(⋃T)×Pow(⋃T). U⊆V″))–is hereditary″
⟨*proof*⟩

theorem (in topology0) anti˙T4:
 shows (T–is anti-″isT4) ⟷ (IsLinOrder(T,–⟨U,V⟩∈Pow(⋃T)×Pow(⋃T). U⊆V″))
⟨*proof*⟩

theorem (in topology0) anti˙T3:
 shows (T–is anti-″isT3) ⟷ (IsLinOrder(T,–⟨U,V⟩∈Pow(⋃T)×Pow(⋃T). U⊆V″))
⟨*proof*⟩

theorem (in topology0) anti˙T2:
 shows (T–is anti-″isT2) ⟷ (IsLinOrder(T,–⟨U,V⟩∈Pow(⋃T)×Pow(⋃T). U⊆V″))
⟨*proof*⟩

lemma linord˙spectrum:
 shows (A–is in the spectrum of″(λT. IsLinOrder(T,–⟨U,V⟩∈Pow(⋃T)×Pow(⋃T). U⊆V″))) ⟷ A≲1
⟨*proof*⟩

theorem (in topology0) anti˙linord:
 shows (T–is anti-″(λT. IsLinOrder(T,–⟨U,V⟩∈Pow(⋃T)×Pow(⋃T). U⊆V″))) ⟷ T–is T_1″
⟨*proof*⟩

In conclusion, T_1 is also an anti-property.

Let's define some anti-properties that we'll use in the future.

definition
 IsAntiComp (˙–is anti-compact″)
 where T–is anti-compact″ ≡ T–is anti-″(λT. (⋃T)–is compact in″T)

definition
 IsAntiLin (˙–is anti-lindeloef″)
 where T–is anti-lindeloef″ ≡ T–is anti-″(λT. ((⋃T)–is lindeloef in″T))

Anti-compact spaces are also called pseudo-finite spaces in literature before the concept of anti-property was defined.

end

61 Topology 6

theory Topology˙ZF˙6 imports Topology˙ZF˙4 Topology˙ZF˙2 Topology˙ZF˙1

begin

This theory deals with the relations between continuous functions and convergence of filters. At the end of the file there some results about the building of functions in cartesian products.

61.1 Image filter

First of all, we will define the appropriate tools to work with functions and filters together.

We define the image filter as the collections of supersets of of images of sets from a filter.

definition
  ImageFilter (˙[˙]..˙ 98)
   where 𝔉 –is a filter on″ X ⟹ f:X→Y ⟹ f[𝔉]..Y ≡ –A∈Pow(Y). ∃D∈–f(B) .B∈𝔉″. D⊆A″

Note that in the previous definition, it is necessary to state Y as the final set because f is also a function to every superset of its range. X can be changed by domain(f) without any change in the definition.

lemma base˙image˙filter:
  assumes 𝔉 –is a filter on″ X f:X→Y
  shows –fB .B∈𝔉″ –is a base filter″ (f[𝔉]..Y) and (f[𝔉]..Y) –is a filter on″ Y
⟨*proof*⟩

61.2 Continuous at a point vs. globally continuous

In this section we show that continuity of a function implies local continuity (at a point) and that local continuity at all points implies (global) continuity.

If a function is continuous, then it is continuous at every point.

lemma cont˙global˙imp˙continuous˙x:
  assumes x∈⋃τ_1 IsContinuous(τ_1,τ_2,f) f:(⋃τ_1)→(⋃τ_2) x∈⋃τ_1
  shows ∀U∈τ_2. f(x)∈U ⟶ (∃V∈τ_1. x∈V ∧ f(V)⊆U)
⟨*proof*⟩

A function that is continuous at every point of its domain is continuous.

lemma ccontinuous˙all˙x˙imp˙cont˙global:
  assumes ∀x∈⋃τ_1. ∀U∈τ_2. fx∈U ⟶ (∃V∈τ_1. x∈V ∧ fV⊆U) f∈(⋃τ_1)→(⋃τ_2) and
    τ_1 –is a topology″
  shows IsContinuous(τ_1,τ_2,f)
⟨*proof*⟩

61.3 Continuous functions and filters

In this section we consider the relations between filters and continuity.

If the function is continuous then if the filter converges to a point the image filter converges to the image point.

lemma (in two˙top˙spaces0) cont˙imp˙filter˙conver˙preserved:
  assumes 𝔉 –is a filter on″ X_1 f –is continuous″ 𝔉 $\rightarrow_F$ x –in″ τ_1

shows (f[$\mathfrak{F}$]..X_2) $\to_F$ (f(x)) –in" τ_2
⟨*proof*⟩

Continuity in filter at every point of the domain implies global continuity.

lemma (in two˙top˙spaces0) filter˙conver˙preserved˙imp˙cont:
assumes $\forall$ x$\in\bigcup\tau_1$. $\forall\mathfrak{F}$. (($\mathfrak{F}$ –is a filter on" X_1) $\wedge$ ($\mathfrak{F}$ $\to_F$ x –in" τ_1)) $\longrightarrow$ ((f[$\mathfrak{F}$]..X_2) $\to_F$ (fx) –in" τ_2)
shows f–is continuous"
⟨*proof*⟩

end

62 Topology 7

theory Topology˙ZF˙7 imports Topology˙ZF˙5
begin

62.1 Connection Properties

Another type of topological properties are the connection properties. These properties establish if the space is formed of several pieces or just one.

A space is connected iff there is no clopen set other that the empty set and the total set.

definition IsConnected (˙–is connected" 70)
where T –is connected" $\equiv$ $\forall$ U. (U$\in$T $\wedge$ (U –is closed in"T)) $\longrightarrow$ U=0$\vee$U=$\bigcup$T

lemma indiscrete˙connected:
shows –0,X" –is connected"
⟨*proof*⟩

The anti-property of connectedness is called total-diconnectedness.

definition IsTotDis (˙ –is totally-disconnected" 70)
where IsTotDis $\equiv$ ANTI(IsConnected)

lemma conn˙spectrum:
shows (A–is in the spectrum of"IsConnected) $\longleftrightarrow$ A$\lesssim$1
⟨*proof*⟩

The discrete space is a first example of totally-disconnected space.

lemma discrete˙tot˙dis:
shows Pow(X) –is totally-disconnected"
⟨*proof*⟩

An space is hyperconnected iff every two non-empty open sets meet.

definition IsHConnected (˙–is hyperconnected"90)
where T–is hyperconnected" $\equiv\forall$ U V. U$\in$T $\wedge$ V$\in$T $\wedge$ U$\cap$V=0 $\longrightarrow$ U=0$\vee$V=0

Every hyperconnected space is connected.

lemma HConn˙imp˙Conn:
 assumes T–is hyperconnected″
 shows T–is connected″
⟨*proof*⟩

lemma Indiscrete˙HConn:
 shows –0,X″–is hyperconnected″
 ⟨*proof*⟩

A first example of an hyperconnected space but not indiscrete, is the cofinite topology on the natural numbers.

lemma Cofinite˙nat˙HConn:
 assumes ¬(X≺nat)
 shows (CoFinite X)–is hyperconnected″
⟨*proof*⟩

lemma HConn˙spectrum:
 shows (A–is in the spectrum of″IsHConnected) ⟷ A≲1
⟨*proof*⟩

In the following results we will show that anti-hyperconnectedness is a separation property between T_1 and T_2. We will show also that both implications are proper.

First, the closure of a point in every topological space is always hyperconnected. This is the reason why every anti-hyperconnected space must be T_1: every singleton must be closed.

lemma (in topology0)cl˙point˙imp˙HConn:
 assumes x∈⋃T
 shows (T–restricted to″Closure(–x″,T))–is hyperconnected″
⟨*proof*⟩

A consequence is that every totally-disconnected space is T_1.

lemma (in topology0) tot˙dis˙imp˙T1:
 assumes T–is totally-disconnected″
 shows T–is T_1″
⟨*proof*⟩

In the literature, there exists a class of spaces called sober spaces; where the only non-empty closed hyperconnected subspaces are the closures of points and closures of different singletons are different.

definition IsSober (˙–is sober″90)
 where T–is sober″ ≡ ∀A∈Pow(⋃T)-–0″. (A–is closed in″T ∧ ((T–restricted to″A)–is hyperconnected″)) ⟶ (∃x∈⋃T. A=Closure(–x″,T) ∧ (∀y∈⋃T. A=Closure(–y″,T) ⟶ y=x))

Being sober is weaker than being anti-hyperconnected.

theorem (in topology0) anti˙HConn˙imp˙sober:
  assumes T–is anti-″IsHConnected
  shows T–is sober″
⟨proof⟩

Every sober space is T_0.

lemma (in topology0) sober˙imp˙T0:
  assumes T–is sober″
  shows T–is T_0″
⟨proof⟩

Every T_2 space is anti-hyperconnected.

theorem (in topology0) T2˙imp˙anti˙HConn:
  assumes T–is T_2″
  shows T–is anti-″IsHConnected
⟨proof⟩

Every anti-hyperconnected space is T_1.

theorem anti˙HConn˙imp˙T1:
  assumes T–is anti-″IsHConnected
  shows T–is T_1″
⟨proof⟩

There is at least one topological space that is T_1, but not anti-hyperconnected. This space is the cofinite topology on the natural numbers.

lemma Cofinite˙not˙anti˙HConn:
  shows ¬((CoFinite nat)–is anti-″IsHConnected) and (CoFinite nat)–is T_1″
⟨proof⟩

The join-topology build from the cofinite topology on the natural numbers, and the excluded set topology on the natural numbers excluding –0,1″; is just the union of both.

lemma join˙top˙cofinite˙excluded˙set:
  shows (joinT –CoFinite nat,ExcludedSet(nat,–0,1″)″)=(CoFinite nat)∪ ExcludedSet(nat,–0,1″)
⟨proof⟩

The previous topology in not T_2, but is anti-hyperconnected.

theorem join˙Cofinite˙ExclPoint˙not˙T2:
  shows
    ¬((joinT –CoFinite nat, ExcludedSet(nat,–0,1″)″)–is T_2″) and
    (joinT –CoFinite nat,ExcludedSet(nat,–0,1″)″)–is anti-″ IsHConnected
⟨proof⟩

Let's show that anti-hyperconnected is in fact T_1 and sober. The trick of the proof lies in the fact that if a subset is hyperconnected, its closure is so too (the closure of a point is then always hyperconnected because singletons are in the spectrum); since the closure is closed, we can apply the sober property on it.

theorem (in topology0) T1˙sober˙imp˙anti˙HConn:
  assumes T–is T_1″ and T–is sober″
  shows T–is anti-″IsHConnected
⟨*proof*⟩

theorem (in topology0) anti˙HConn˙iff˙T1˙sober:
  shows (T–is anti-″IsHConnected) ⟷ (T–is sober″∧T–is T_1″)
  ⟨*proof*⟩

A space is ultraconnected iff every two non-empty closed sets meet.

definition IsUConnected (˙–is ultraconnected″80)
  where T–is ultraconnected″≡ ∀ A B. A–is closed in″T∧B–is closed in″T∧A∩B=0 ⟶ A=0∨B=0

Every ultraconnected space is trivially normal.

lemma (in topology0)UConn˙imp˙normal:
  assumes T–is ultraconnected″
  shows T–is normal″
⟨*proof*⟩

Every ultraconnected space is connected.

lemma UConn˙imp˙Conn:
  assumes T–is ultraconnected″
  shows T–is connected″
⟨*proof*⟩

lemma UConn˙spectrum:
  shows (A–is in the spectrum of″IsUConnected) ⟷ A$\lesssim$1
⟨*proof*⟩

This time, anti-ultraconnected is an old property.

theorem (in topology0) anti˙UConn:
  shows (T–is anti-″IsUConnected) ⟷ T–is T_1″
⟨*proof*⟩

Is is natural that separation axioms and connection axioms are anti-properties of each other; as the concepts of connectedness and separation are opposite.

To end this section, let's try to charaterize anti-sober spaces.

lemma sober˙spectrum:
  shows (A–is in the spectrum of″IsSober) ⟷ A$\lesssim$1
⟨*proof*⟩

theorem (in topology0)anti˙sober:
  shows (T–is anti-″IsSober) ⟷ T=–0,$\bigcup$T″

⟨*proof*⟩

end

63 Topology 8

theory Topology˙ZF˙8 imports Topology˙ZF˙6 EquivClass1
begin

This theory deals with quotient topologies.

63.1 Definition of quotient topology

Given a surjective function $f : X \to Y$ and a topology τ in X, it is posible to consider a special topology in Y. f is called quotient function.

definition(in topology0)
  QuotientTop (–quotient topology in″˙–by″˙ 80)
  where f∈surj(⋃T,Y) ⟹–quotient topology in″Y–by″f≡
    –U∈Pow(Y). f-U∈T″

abbreviation QuotientTopTop (–quotient topology in″˙–by″˙–from″˙)
  where QuotientTopTop(Y,f,T) ≡ topology0.QuotientTop(T,Y,f)

The quotient topology is indeed a topology.

theorem(in topology0) quotientTop˙is˙top:
  assumes f∈surj(⋃T,Y)
  shows (–quotient topology in″ Y –by″ f) –is a topology″
⟨*proof*⟩

The quotient function is continuous.

lemma (in topology0) quotient˙func˙cont:
  assumes f∈surj(⋃T,Y)
  shows IsContinuous(T,(–quotient topology in″ Y –by″ f),f)
    ⟨*proof*⟩

One of the important properties of this topology, is that a function from the quotient space is continuous iff the composition with the quotient function is continuous.

theorem(in two˙top˙spaces0) cont˙quotient˙top:
  assumes h∈surj(⋃τ_1,Y) g:Y→⋃τ_2 IsContinuous(τ_1,τ_2,g O h)
  shows IsContinuous((–quotient topology in″ Y –by″ h –from″ τ_1),τ_2,g)
⟨*proof*⟩

The underlying set of the quotient topology is Y.

lemma(in topology0) total˙quo˙func:

assumes f∈surj(⋃T,Y)
shows (⋃(–quotient topology in″Y–by″f))=Y
⟨*proof*⟩

63.2 Quotient topologies from equivalence relations

In this section we will show that the quotient topologies come from an equivalence relation.

First, some lemmas for relations.

lemma quotient˙proj˙fun:
shows –⟨b,r–b″⟩. b∈A″:A→A//r ⟨*proof*⟩

lemma quotient˙proj˙surj:
shows –⟨b,r–b″⟩. b∈A″∈surj(A,A//r)
⟨*proof*⟩

lemma preim˙equi˙proj:
assumes U⊆A//r equiv(A,r)
shows –⟨b,r–b″⟩. b∈A″-U=⋃U
⟨*proof*⟩

Now we define what a quotient topology from an equivalence relation is:

definition(in topology0)
EquivQuo (–quotient by″˙ 70)
where equiv(⋃T,r)⟹(–quotient by″r)≡–quotient topology in″(⋃T)//r–by″–⟨b,r–b″⟩. b∈⋃T″

abbreviation
EquivQuoTop (˙–quotient by″˙ 60)
where EquivQuoTop(T,r)≡topology0.EquivQuo(T,r)

First, another description of the topology (more intuitive):

theorem (in topology0) quotient˙equiv˙rel:
assumes equiv(⋃T,r)
shows (–quotient by″r)=–U∈Pow((⋃T)//r). ⋃U∈T″
⟨*proof*⟩

We apply previous results to this topology.

theorem(in topology0) total˙quo˙equi:
assumes equiv(⋃T,r)
shows ⋃(–quotient by″r)=(⋃T)//r
⟨*proof*⟩

theorem(in topology0) equiv˙quo˙is˙top:
assumes equiv(⋃T,r)
shows (–quotient by″r)–is a topology″
⟨*proof*⟩

MAIN RESULT: All quotient topologies arise from an equivalence relation given by the quotient function $f : X \to Y$. This means that any quotient topology is homeomorphic to a topology given by an equivalence relation quotient.

theorem(in topology0) equiv˙quotient˙top:
  assumes f∈surj(⋃T,Y)
  defines r≡–⟨x,y⟩∈⋃T×⋃T. f(x)=f(y)″
  defines g≡–⟨y,f-–y″⟩. y∈Y″
 shows equiv(⋃T,r) and IsAhomeomorphism((–quotient topology in″Y–by″f),(–quotient by″r),g)
⟨*proof*⟩

lemma product˙equiv˙rel˙fun:
  shows –⟨⟨b,c⟩,⟨r–b″,r–c″⟩⟩. ⟨b,c⟩∈⋃T×⋃T″:(⋃T×⋃T)→((⋃T)//r×(⋃T)//r)
⟨*proof*⟩

lemma(in topology0) prod˙equiv˙rel˙surj:
 shows –⟨⟨b,c⟩,⟨r–b″,r–c″⟩⟩. ⟨b,c⟩∈⋃T×⋃T″:surj(⋃(ProductTopology(T,T)),((⋃T)//r×(⋃T)//r))
⟨*proof*⟩

lemma(in topology0) product˙quo˙fun:
  assumes equiv(⋃T,r)
 shows IsContinuous(ProductTopology(T,T),ProductTopology(–quotient by″r,(–quotient by″r)),–⟨⟨b,c⟩,⟨r–b″,r–c″⟩⟩. ⟨b,c⟩∈⋃T×⋃T″)
⟨*proof*⟩

The product of quotient topologies is a quotient topology given that the quotient map is open. This isn't true in general.

theorem(in topology0) prod˙quotient:
  assumes equiv(⋃T,r) ∀A∈T. –⟨b,r–b″⟩. b∈⋃T″A∈(–quotient by″r)
  shows (ProductTopology(–quotient by″r,–quotient by″r)) = (–quotient topology in″(((⋃T)//r)×((⋃T)//r))–by″(–⟨⟨b,c⟩,⟨r–b″,r–c″⟩⟩. ⟨b,c⟩∈⋃T×⋃T″)–from″(ProductTopology(T,T)))
⟨*proof*⟩

end

64 Topology 9

theory Topology˙ZF˙9
imports Topology˙ZF˙2 Group˙ZF˙2 Topology˙ZF˙7 Topology˙ZF˙8
begin

64.1 Group of homeomorphisms

This theory file deals with the fact the set homeomorphisms of a topological space into itself forms a group.

First, we define the set of homeomorphisms.

definition
  HomeoG(T) ≡ –f:⋃T→⋃T. IsAhomeomorphism(T,T,f)˝

The homeomorphisms are closed by composition.

lemma (in topology0) homeo˙composition:
  assumes f∈HomeoG(T)g∈HomeoG(T)
  shows Composition(⋃T)⟨f, g⟩∈HomeoG(T)
⟨*proof*⟩

The identity function is a homeomorphism.

lemma (in topology0) homeo˙id:
  shows id(⋃T)∈HomeoG(T)
⟨*proof*⟩

The homeomorphisms form a monoid and its neutral element is the identity.

theorem (in topology0) homeo˙submonoid:
 shows IsAmonoid(HomeoG(T),restrict(Composition(⋃T),HomeoG(T)×HomeoG(T)))

 TheNeutralElement(HomeoG(T),restrict(Composition(⋃T),HomeoG(T)×HomeoG(T)))=id(⋃T)
⟨*proof*⟩

The homeomorphisms form a group, with the composition.

theorem(in topology0) homeo˙group:
 shows IsAgroup(HomeoG(T),restrict(Composition(⋃T),HomeoG(T)×HomeoG(T)))
⟨*proof*⟩

64.2 Examples computed

As a first example, we show that the group of homeomorphisms of the cocardinal topology is the group of bijective functions.

theorem homeo˙cocardinal:
  assumes InfCard(Q)
  shows HomeoG(CoCardinal(X,Q))=bij(X,X)
⟨*proof*⟩

The group of homeomorphism of the excluded set is a direct product of the bijections on $X \setminus T$ and the bijections on $X \cap T$.

theorem homeo˙excluded:
  shows HomeoG(ExcludedSet(X,T))=–f∈bij(X,X). f(X-T)=(X-T)˝
⟨*proof*⟩

We now give some lemmas that will help us compute HomeoG(IncludedSet(X,T)).

lemma cont˙in˙cont˙ex:
  assumes IsContinuous(IncludedSet(X,T),IncludedSet(X,T),f) f:X→X T⊆X
  shows IsContinuous(ExcludedSet(X,T),ExcludedSet(X,T),f)
⟨*proof*⟩

lemma cont˙ex˙cont˙in:
assumes IsContinuous(ExcludedSet(X,T),ExcludedSet(X,T),f) $f:X\to X$ $T\subseteq X$
shows IsContinuous(IncludedSet(X,T),IncludedSet(X,T),f)
⟨*proof*⟩

The previous lemmas imply that the group of homeomorphisms of the included set topology is the same as the one of the excluded set topology.

lemma homeo˙included:
assumes $T\subseteq X$
shows HomeoG(IncludedSet(X,T))=–f ∈ bij(X, X) . f (X - T) = X - T″
⟨*proof*⟩

Finally, let's compute part of the group of homeomorphisms of an order topology.

lemma homeo˙order:
assumes IsLinOrder(X,r)$\exists$ x y. $x\neq y\wedge x\in X\wedge y\in X$
shows ord˙iso(X,r,X,r)$\subseteq$HomeoG(OrdTopology X r)
⟨*proof*⟩

This last example shows that order isomorphic sets give homeomorphic topological spaces.

64.3 Properties preserved by functions

The continuous image of a connected space is connected.

theorem (in two˙top˙spaces0) cont˙image˙conn:
assumes IsContinuous(τ_1,τ_2,f) f$\in$surj(X_1,X_2) τ_1–is connected″
shows τ_2–is connected″
⟨*proof*⟩

Every continuous function from a space which has some property P and a space which has the property anti(P), given that this property is preserved by continuous functions, if follows that the range of the function is in the spectrum. Applied to connectedness, it follows that continuous functions from a connected space to a totally-disconnected one are constant.

corollary(in two˙top˙spaces0) cont˙conn˙tot˙disc:
assumes IsContinuous(τ_1,τ_2,f) τ_1–is connected″ τ_2–is totally-disconnected″ $f:X_1\to X_2$ $X_1\neq 0$
shows $\exists$ q$\in X_2$. $\forall$ w$\in X_1$. f(w)=q
⟨*proof*⟩

The continuous image of a compact space is compact.

theorem (in two˙top˙spaces0) cont˙image˙com:
assumes IsContinuous(τ_1,τ_2,f) f$\in$surj(X_1,X_2) X_1–is compact of cardinal″K–in″τ_1
shows X_2–is compact of cardinal″K–in″τ_2
⟨*proof*⟩

As it happends to connected spaces, a continuous function from a compact space to an anti-compact space has finite range.

corollary (in two˙top˙spaces0) cont˙comp˙anti˙comp:
 assumes IsContinuous(τ_1,τ_2,f) X_1–is compact in″τ_1 τ_2–is anti-compact″ f:$X_1 \rightarrow X_2$ $X_1 \neq 0$
 shows Finite(range(f)) and range(f)$\neq$0
⟨*proof*⟩

As a consequence, it follows that quotient topological spaces of compact (connected) spaces are compact (connected).

corollary(in topology0) compQuot:
 assumes ($\bigcup$T)–is compact in″T equiv($\bigcup$T,r)
 shows ($\bigcup$T)//r–is compact in″(–quotient by″r)
⟨*proof*⟩

corollary(in topology0) ConnQuot:
 assumes T–is connected″ equiv($\bigcup$T,r)
 shows (–quotient by″r)–is connected″
⟨*proof*⟩

end

65 Topology 10

theory Topology˙ZF˙10
imports Topology˙ZF˙7
begin

This file deals with properties of product spaces. We only consider product of two spaces, and most of this proofs, can be used to prove the results in product of a finite number of spaces.

65.1 Closure and closed sets in product space

The closure of a product, is the product of the closures.

lemma cl˙product:
 assumes T–is a topology″ S–is a topology″ A$\subseteq\bigcup$T B$\subseteq\bigcup$S
 shows Closure(A$\times$B,ProductTopology(T,S))=Closure(A,T)$\times$Closure(B,S)
⟨*proof*⟩

The product of closed sets, is closed in the product topology.

corollary closed˙product:
 assumes T–is a topology″ S–is a topology″ A–is closed in″TB–is closed in″S
 shows (A$\times$B) –is closed in″ProductTopology(T,S)
⟨*proof*⟩

65.2 Separation properties in product space

The product of T_0 spaces is T_0.

theorem T0˙product:
  assumes T–is a topology″S–is a topology″T–is T_0″S–is T_0″
  shows ProductTopology(T,S)–is T_0″
⟨*proof*⟩

The product of T_1 spaces is T_1.

theorem T1˙product:
  assumes T–is a topology″S–is a topology″T–is T_1″S–is T_1″
  shows ProductTopology(T,S)–is T_1″
⟨*proof*⟩

The product of T_2 spaces is T_2.

theorem T2˙product:
  assumes T–is a topology″S–is a topology″T–is T_2″S–is T_2″
  shows ProductTopology(T,S)–is T_2″
⟨*proof*⟩

The product of regular spaces is regular.

theorem regular˙product:
  assumes T–is a topology″ S–is a topology″ T–is regular″ S–is regular″
  shows ProductTopology(T,S)–is regular″
⟨*proof*⟩

65.3 Connection properties in product space

First, we prove that the projection functions are open.

lemma projection˙open:
  assumes T–is a topology″S–is a topology″B∈ProductTopology(T,S)
  shows –y∈$\bigcup$T. ∃x∈$\bigcup$S. ⟨y,x⟩∈B″∈T
⟨*proof*⟩

lemma projection˙open2:
  assumes T–is a topology″S–is a topology″B∈ProductTopology(T,S)
  shows –y∈$\bigcup$S. ∃x∈$\bigcup$T. ⟨x,y⟩∈B″∈S
⟨*proof*⟩

The product of connected spaces is connected.

theorem compact˙product:
  assumes T–is a topology″S–is a topology″T–is connected″S–is connected″
  shows ProductTopology(T,S)–is connected″
⟨*proof*⟩

end

66 Topology 11

theory Topology˙ZF˙11 imports Topology˙ZF˙7 Finite˙ZF˙1

begin

This file deals with order topologies. The order topology is already defined in Topology˙ZF˙examples˙1.thy.

66.1 Order topologies

We will assume most of the time that the ordered set has more than one point. It is natural to think that the topological properties can be translated to properties of the order; since every order rises one and only one topology in a set.

66.2 Separation properties

Order topologies have a lot of separation properties.

Every order topology is Hausdorff.

theorem order˙top˙T2:
  assumes IsLinOrder(X,r) ∃x y. x≠y∧x∈X∧y∈X
  shows (OrdTopology X r)–is T_2″
⟨*proof*⟩

Every order topology is T_4, but the proof needs lots of machinery. At the end of the file, we will prove that every order topology is normal; sooner or later.

66.3 Connectedness properties

Connectedness is related to two properties of orders: completeness and density

Some order-dense properties:

definition
  IsDenseSub (˙ –is dense in″˙–with respect to″˙) where
  A –is dense in″X–with respect to″r ≡
  ∀x∈X. ∀y∈X. ⟨x,y⟩∈r ∧ x≠y ⟶ (∃z∈A-–x,y″. ⟨x,z⟩∈r∧⟨z,y⟩∈r)

definition
  IsDenseUnp (˙ –is not-properly dense in″˙–with respect to″˙) where
  A –is not-properly dense in″X–with respect to″r ≡
  ∀x∈X. ∀y∈X. ⟨x,y⟩∈r ∧ x≠y ⟶ (∃z∈A. ⟨x,z⟩∈r∧⟨z,y⟩∈r)

definition
IsWeaklyDenseSub (˙ –is weakly dense in˝˙–with respect to˝˙) where
A –is weakly dense in˝X–with respect to˝r ≡
∀x∈X. ∀y∈X. ⟨x,y⟩∈r ∧ x≠y ⟶ ((∃z∈A-–x,y˝. ⟨x,z⟩∈r∧⟨z,y⟩∈r)∨ IntervalX(X,r,x,y)=0)

definition
IsDense (˙ –is dense with respect to˝˙) where
X –is dense with respect to˝r ≡
∀x∈X. ∀y∈X. ⟨x,y⟩∈r ∧ x≠y ⟶ (∃z∈X-–x,y˝. ⟨x,z⟩∈r∧⟨z,y⟩∈r)

lemma dense˙sub:
shows (X –is dense with respect to˝r) ⟷ (X –is dense in˝X–with respect to˝r)
⟨*proof*⟩

lemma not˙prop˙dense˙sub:
shows (A –is dense in˝X–with respect to˝r) ⟶ (A –is not-properly dense in˝X–with respect to˝r)
⟨*proof*⟩

In densely ordered sets, intervals are infinite.

theorem dense˙order˙inf˙intervals:
assumes IsLinOrder(X,r) IntervalX(X, r, b, c)≠0b∈Xc∈X X–is dense with respect to˝r
shows ¬Finite(IntervalX(X, r, b, c))
⟨*proof*⟩

Left rays are infinite.

theorem dense˙order˙inf˙lrays:
assumes IsLinOrder(X,r) LeftRayX(X,r,c)≠0c∈X X–is dense with respect to˝r
shows ¬Finite(LeftRayX(X,r,c))
⟨*proof*⟩

Right rays are infinite.

theorem dense˙order˙inf˙rrays:
assumes IsLinOrder(X,r) RightRayX(X,r,b)≠0b∈X X–is dense with respect to˝r
shows ¬Finite(RightRayX(X,r,b))
⟨*proof*⟩

The whole space in a densely ordered set is infinite.

corollary dense˙order˙infinite:
assumes IsLinOrder(X,r) X–is dense with respect to˝r
∃x y. x≠y∧x∈X∧y∈X
shows ¬(X≺nat)
⟨*proof*⟩

If an order topology is connected, then the order is complete. It is equivalent to assume that $r \subseteq X \times X$ or prove that $r \cap X \times X$ is complete.

theorem conn˙imp˙complete:

assumes IsLinOrder(X,r) ∃x y. x≠y∧x∈X∧y∈X r⊆X×X
(OrdTopology X r)–is connected"
shows r–is complete"
⟨*proof*⟩

If an order topology is connected, then the order is dense.

theorem conn˙imp˙dense:
assumes IsLinOrder(X,r) ∃x y. x≠y∧x∈X∧y∈X
(OrdTopology X r)–is connected"
shows X –is dense with respect to"r
⟨*proof*⟩

Actually a connected order topology is one that comes from a dense and complete order.

First a lemma. In a complete ordered set, every non-empty set bounded from below has a maximum lower bound.

lemma complete˙order˙bounded˙below:
assumes r–is complete" IsBoundedBelow(A,r) A≠0 r⊆X×X
shows HasAmaximum(r,⋂c∈A. r-–c")
⟨*proof*⟩

theorem comp˙dense˙imp˙conn:
assumes IsLinOrder(X,r) ∃x y. x≠y∧x∈X∧y∈X r⊆X×X
X –is dense with respect to"r r–is complete"
shows (OrdTopology X r)–is connected"
⟨*proof*⟩

66.4 Numerability axioms

A κ-separable order topology is in relation with order density.

If an order topology has a subset A which is topologically dense, then that subset is weakly order-dense in X.

lemma dense˙top˙imp˙Wdense˙ord:
assumes IsLinOrder(X,r) Closure(A,OrdTopology X r)=X A⊆X ∃x y. x ≠ y ∧ x ∈ X ∧ y ∈ X
shows A–is weakly dense in"X–with respect to"r
⟨*proof*⟩

Conversely, a weakly order-dense set is topologically dense if it is also considered that: if there is a maximum or a minimum elements whose singletons are open, this points have to be in A. In conclusion, weakly order-density is a property closed to topological density.

Another way to see this: Consider a weakly order-dense set A:

- If X has a maximum and a minimum and $\{min, max\}$ is open: A is topologically dense in $X \setminus \{min, max\}$, where min is the minimum in X and max is the maximum in X.

- If X has a maximum, $\{max\}$ is open and X has no minimum or $\{min\}$ isn't open: A is topologically dense in $X \setminus \{max\}$, where max is the maximum in X.

- If X has a minimum, $\{min\}$ is open and X has no maximum or $\{max\}$ isn't open A is topologically dense in $X \setminus \{min\}$, where min is the minimum in X.

- If X has no minimum or maximum, or $\{min, max\}$ has no proper open sets: A is topologically dense in X.

lemma Wdense˙ord˙imp˙dense˙top:
  assumes IsLinOrder(X,r) A–is weakly dense in˝X–with respect to˝r A⊆X ∃x y. x ≠ y ∧ x ∈ X ∧ y ∈ X
   HasAminimum(r,X)⟶–Minimum(r,X)˝∈(OrdTopology X r)⟶Minimum(r,X)∈A
   HasAmaximum(r,X)⟶–Maximum(r,X)˝∈(OrdTopology X r)⟶Maximum(r,X)∈A
  shows Closure(A,OrdTopology X r)=X
⟨*proof*⟩

The conclusion is that an order topology is κ-separable iff there is a set A with cardinality strictly less than κ which is weakly-dense in X.

theorem separable˙imp˙wdense:
  assumes (OrdTopology X r)–is separable of cardinal˝Q ∃x y. x ≠ y ∧ x ∈ X ∧ y ∈ X
    IsLinOrder(X,r)
  shows ∃A∈Pow(X). A≺Q ∧ (A–is weakly dense in˝X–with respect to˝r)
⟨*proof*⟩

theorem wdense˙imp˙separable:
  assumes ∃x y. x ≠ y ∧ x ∈ X ∧ y ∈ X (A–is weakly dense in˝X–with respect to˝r)
    IsLinOrder(X,r) A≺Q InfCard(Q) A⊆X
  shows (OrdTopology X r)–is separable of cardinal˝Q
⟨*proof*⟩

end

67 Properties in topology 2

theory Topology˙ZF˙properties˙2 imports Topology˙ZF˙7 Topology˙ZF˙1b
  Finite˙ZF˙1 Topology˙ZF˙11

begin

67.1 Local properties.

This theory file deals with local topological properties; and applies local compactness to the one point compactification.

We will say that a topological space is locally @term"P" iff every point has a neighbourhood basis of subsets that have the property @term"P" as subspaces.

definition
 IsLocally (˙–is locally˝˙ 90)
 where T–is a topology˝ $\Longrightarrow$ T–is locally˝P ≡ (∀ x∈⋃ T. ∀ b∈T. x∈b ⟶ (∃ c∈Pow(b). x∈Interior(c,T) ∧ P(c,T)))

67.2 First examples

Our first examples deal with the locally finite property. Finiteness is a property of sets, and hence it is preserved by homeomorphisms; which are in particular bijective.

The discrete topology is locally finite.

lemma discrete˙locally˙finite:
 shows Pow(A)–is locally˝(λA.(λB. Finite(A)))
⟨*proof*⟩

The included set topology is locally finite when the set is finite.

lemma included˙finite˙locally˙finite:
 assumes Finite(A) and A⊆X
 shows (IncludedSet(X,A))–is locally˝(λA.(λB. Finite(A)))
⟨*proof*⟩

67.3 Local compactness

definition
 IsLocallyComp (˙–is locally-compact˝ 70)
 where T–is locally-compact˝≡T–is locally˝(λB. λT. B–is compact in˝T)

We center ourselves in local compactness, because it is a very important tool in topological groups and compactifications.

If a subset is compact of some cardinal for a topological space, it is compact of the same cardinal in the subspace topology.

lemma compact˙imp˙compact˙subspace:
 assumes A–is compact of cardinal˝K–in˝T A⊆B
 shows A–is compact of cardinal˝K–in˝(T–restricted to˝B) ⟨*proof*⟩

The converse of the previous result is not always true. For compactness, it holds because the axiom of finite choice always holds.

lemma compact˙subspace˙imp˙compact:
assumes A–is compact in″(T–restricted to″B) A⊆B
shows A–is compact in″T ⟨*proof*⟩

If the axiom of choice holds for some cardinal, then we can drop the compact sets of that cardial are compact of the same cardinal as subspaces of every superspace.

lemma Kcompact˙subspace˙imp˙Kcompact:
assumes A–is compact of cardinal″Q–in″(T–restricted to″B) A⊆B (–the axiom of″ Q –choice holds″)
shows A–is compact of cardinal″Q–in″T
⟨*proof*⟩

Every set, with the cofinite topology is compact.

lemma cofinite˙compact:
shows X –is compact in″(CoFinite X) ⟨*proof*⟩

A corollary is then that the cofinite topology is locally compact; since every subspace of a cofinite space is cofinite.

corollary cofinite˙locally˙compact:
shows (CoFinite X)–is locally-compact″
⟨*proof*⟩

In every locally compact space, by definition, every point has a compact neighbourhood.

theorem (in topology0) locally˙compact˙exist˙compact˙neig:
assumes T–is locally-compact″
shows ∀x∈⋃T. ∃A∈Pow(⋃T). A–is compact in″T ∧ x∈int(A)
⟨*proof*⟩

In Hausdorff spaces, the previous result is an equivalence.

theorem (in topology0) exist˙compact˙neig˙T2˙imp˙locally˙compact:
assumes ∀x∈⋃T. ∃A∈Pow(⋃T). x∈int(A) ∧ A–is compact in″T T–is T_2″
shows T–is locally-compact″
⟨*proof*⟩

67.4 Compactification by one point

Given a topological space, we can always add one point to the space and get a new compact topology; as we will check in this section.

definition
OPCompactification (–one-point compactification of″˙ 90)
where –one-point compactification of″T≡T∪—⋃T″∪((⋃T)-K). K∈–B∈Pow(⋃T). B–is compact in″T ∧ B–is closed in″T″″

Firstly, we check that what we defined is indeed a topology.

theorem (in topology0) op˙comp˙is˙top:

shows (–one-point compactification of″T)–is a topology″ ⟨*proof*⟩

The original topology is an open subspace of the new topology.

theorem (in topology0) open˙subspace:
shows $\bigcup$T∈–one-point compactification of″T and (–one-point compactification of″T)–restricted to″$\bigcup$T=T
⟨*proof*⟩

We added only one new point to the space.

lemma (in topology0) op˙compact˙total:
shows $\bigcup$(–one-point compactification of″T)=–$\bigcup$T″∪($\bigcup$T)
⟨*proof*⟩

The one point compactification, gives indeed a compact topological space.

theorem (in topology0) compact˙op:
shows (–$\bigcup$T″∪($\bigcup$T))–is compact in″(–one-point compactification of″T) ⟨*proof*⟩

The one point compactification is Hausdorff iff the original space is also Hausdorff and locally compact.

lemma (in topology0) op˙compact˙T2˙1:
assumes (–one-point compactification of″T)–is T_2″
shows T–is T_2″
⟨*proof*⟩

lemma (in topology0) op˙compact˙T2˙2:
assumes (–one-point compactification of″T)–is T_2″
shows T–is locally-compact″
⟨*proof*⟩

lemma (in topology0) op˙compact˙T2˙3:
assumes T–is locally-compact″ T–is T_2″
shows (–one-point compactification of″T)–is T_2″
⟨*proof*⟩

In conclusion, every locally compact Hausdorff topological space is regular; since this property is hereditary.

corollary (in topology0) locally˙compact˙T2˙imp˙regular:
assumes T–is locally-compact″ T–is T_2″
shows T–is regular″
⟨*proof*⟩

This last corollary has an explanation: In Hausdorff spaces, compact sets are closed and regular spaces are exactly the "locally closed spaces"(those which have a neighbourhood basis of closed sets). So the neighbourhood basis of compact sets also works as the neighbourhood basis of closed sets we needed to find.

definition

IsLocallyClosed (˙–is locally-closed˝)
where T–is locally-closed˝ ≡ T–is locally˝(λB TT. B–is closed in˝TT)

lemma (in topology0) regular˙locally˙closed:
shows T–is regular˝ ⟷ (T–is locally-closed˝)
⟨*proof*⟩

67.5 Hereditary properties and local properties

In this section, we prove a relation between a property and its local property for hereditary properties. Then we apply it to locally-Hausdorff or locally-T_2. We also prove the relation between locally-T_2 and another property that appeared when considering anti-properties, the anti-hyperconnectness.

If a property is hereditary in open sets, then local properties are equivalent to find just one open neighbourhood with that property instead of a whole local basis.

lemma (in topology0) her˙P˙is˙loc˙P:
assumes ∀TT. ∀B∈Pow(⋃TT). ∀A∈TT. TT–is a topology˝∧P(B,TT) ⟶ P(B∩A,TT)
shows (T–is locally˝P) ⟷ (∀x∈⋃T. ∃A∈T. x∈A∧P(A,T))
⟨*proof*⟩

definition
IsLocallyT2 (˙–is locally-T_2˝ 70)
where T–is locally-T_2˝≡T–is locally˝(λB. λT. (T–restricted to˝B)–is T_2˝)

Since T_2 is an hereditary property, we can apply the previous lemma.

corollary (in topology0) loc˙T2:
shows (T–is locally-T_2˝) ⟷ (∀x∈⋃T. ∃A∈T. x∈A∧(T–restricted to˝A)–is T_2˝)
⟨*proof*⟩

First, we prove that a locally-T_2 space is anti-hyperconnected.

Before starting, let's prove that an open subspace of an hyperconnected space is hyperconnected.

lemma(in topology0) open˙subspace˙hyperconn:
assumes T–is hyperconnected˝ U∈T
shows (T–restricted to˝U)–is hyperconnected˝
⟨*proof*⟩

lemma(in topology0) locally˙T2˙is˙antiHConn:
assumes T–is locally-T_2˝
shows T–is anti-˝IsHConnected
⟨*proof*⟩

Now we find a counter-example for: Every anti-hyperconnected space is locally-Hausdorff.

The example we are going to consider is the following. Put in X an anti-hyperconnected topology, where an infinite number of points don't have finite sets as neighbourhoods. Then add a new point to the set, $p \notin X$. Consider the open sets on $X \cup p$ as the anti-hyperconnected topology and the open sets that contain p are $p \cup A$ where $X \setminus A$ is finite.

This construction equals the one-point compactification iff X is anti-compact; i.e., the only compact sets are the finite ones. In general this topology is contained in the one-point compactification topology, making it compact too.

It is easy to check that any open set containing p meets infinite other non-empty open set. The question is if such a topology exists.

theorem (in topology0) COF˙comp˙is˙top:
 assumes T–is T_1″¬(⋃T≺nat)
 shows (((–one-point compactification of″(CoFinite (⋃T)))-—⋃T″″)∪T) –is a topology″
⟨*proof*⟩

The previous construction preserves anti-hyperconnectedness.

theorem (in topology0) COF˙comp˙antiHConn:
 assumes T–is anti-″IsHConnected ¬(⋃T≺nat)
 shows (((–one-point compactification of″(CoFinite (⋃T)))-—⋃T″″)∪T) –is anti-″IsHConnected
⟨*proof*⟩

The previous construction, applied to a densely ordered topology, gives the desired counterexample. What happends is that every neighbourhood of ⋃T is dense; because there are no finite open sets, and hence meets every non-empty open set. In conclusion, ⋃T cannot be separated from other points by disjoint open sets.

Every open set that contains ⋃T is dense, when considering the order topology in a densely ordered set with more than two points.

theorem neigh˙infPoint˙dense:
 fixes T X r
 defines T˙def:T ≡ (OrdTopology X r)
 assumes IsLinOrder(X,r) X–is dense with respect to″r
 ∃x y. x≠y∧x∈X∧y∈X U∈((–one-point compactification of″(CoFinite (⋃T)))-—⋃T″″)∪T
⋃T∈U
 V∈((–one-point compactification of″(CoFinite (⋃T)))-—⋃T″″)∪T V≠0
 shows U∩V≠0
⟨*proof*⟩

A densely ordered set with more than one point gives an order topology. Applying the previous construction to this topology we get a non locally-Hausdorff space.

theorem OPComp˙cofinite˙dense˙order˙not˙loc˙T2:

fixes T X r
defines T˙def:T ≡ (OrdTopology X r)
assumes IsLinOrder(X,r) X–is dense with respect to˝r
∃x y. x≠y∧x∈X∧y∈X
shows ¬(((–one-point compactification of˝(CoFinite (⋃T)))-—⋃T˝˝∪T)–is locally-T₂˝)
⟨proof⟩

This topology, from the previous result, gives a counter-example for anti-hyperconnected implies locally-T_2.

theorem antiHConn˙not˙imp˙loc˙T2:
fixes T X r
defines T˙def:T ≡ (OrdTopology X r)
assumes IsLinOrder(X,r) X–is dense with respect to˝r
∃x y. x≠y∧x∈X∧y∈X
shows ¬(((–one-point compactification of˝(CoFinite (⋃T)))-—⋃T˝˝∪T)–is locally-T₂˝)
and ((–one-point compactification of˝(CoFinite (⋃T)))-—⋃T˝˝∪T)–is anti-˝IsHConnected
⟨proof⟩

Let's prove that T_2 spaces are locally-T_2, but that there are locally-T_2 spaces which aren't T_2. In conclusion $T_2 \Rightarrow$ locally -$T_2 \Rightarrow$ anti-hyperconnected; all implications proper.

theorem(in topology0) T2˙imp˙loc˙T2:
assumes T–is T₂˝
shows T–is locally-T₂˝
⟨proof⟩

If there is a closed singleton, then we can consider a topology that makes this point doble.

theorem(in topology0) doble˙point˙top:
assumes –m˝–is closed in˝T
shows (T ∪–(U-–m˝)∪–⋃T˝∪W. ⟨U,W⟩∈–V∈T. m∈V˝×T˝) –is a topology˝
⟨proof⟩

The previous topology is defined over a set with one more point.

lemma(in topology0) union˙doublepoint˙top:
assumes –m˝–is closed in˝T
shows ⋃(T∪–(U-–m˝)∪–⋃T˝∪W. ⟨U,W⟩∈–V∈T. m∈V˝×T˝)=⋃T ∪–⋃T˝
⟨proof⟩

In this topology, the previous topological space is an open subspace.

theorem(in topology0) open˙subspace˙double˙point:
assumes –m˝–is closed in˝T
shows (T∪–(U-–m˝)∪–⋃T˝∪W. ⟨U,W⟩∈–V∈T. m∈V˝×T˝)–restricted to˝⋃T=T
and ⋃T∈(T∪–(U-–m˝)∪–⋃T˝∪W. ⟨U,W⟩∈–V∈T. m∈V˝×T˝)
⟨proof⟩

The previous topology construction applied to a T_2 non-discrite space topology, gives a counter-example to: Every locally-T_2 space is T_2.

If there is a singleton which is not open, but closed; then the construction on that point is not T_2.

theorem(in topology0) loc˙T2˙imp˙T2˙counter˙1:
  assumes –m″∉T –m″–is closed in″T
  shows ¬((T∪–(U-–m″)∪–⋃T″∪W. ⟨U,W⟩∈–V∈T. m∈V″×T″) –is T_2″)
⟨*proof*⟩

This topology is locally-T_2.

theorem(in topology0) loc˙T2˙imp˙T2˙counter˙2:
  assumes –m″∉T m∈⋃T T–is T_2″
  shows (T∪–(U-–m″)∪–⋃T″∪W. ⟨U,W⟩∈–V∈T. m∈V″×T″) –is locally-T_2″
⟨*proof*⟩

There can be considered many more local properties, which; as happens with locally-T_2; can distinguish between spaces other properties cannot.

end

68 Properties in Topology 3

theory Topology˙ZF˙properties˙3 imports Topology˙ZF˙7 Finite˙ZF˙1 Topology˙ZF˙1b Topology˙ZF˙9
  Topology˙ZF˙properties˙2 FinOrd˙ZF
begin

This theory file deals with more topological properties and the relation with the previous ones in other theory files.

68.1 More anti-properties

In this section we study more anti-properties.

68.2 First examples

A first example of an anti-compact space is the discrete space.

lemma pow˙compact˙imp˙finite:
  assumes B–is compact in″Pow(A)
  shows Finite(B)
⟨*proof*⟩

theorem pow˙anti˙compact:
  shows Pow(A)–is anti-compact″
⟨*proof*⟩

In a previous file, Topology˙ZF˙5.thy, we proved that the spectrum of the lindelf property depends on the axiom of countable choice on subsets of the power set of the natural number.

In this context, the examples depend on wether this choice principle holds or not. This is the reason that the examples of anti-lindeloef topologies are left for the next section.

68.3 Structural results

We first differenciate the spectrum of the lindeloef property depending on some axiom of choice.

lemma lindeloef˙spec1:
 assumes –the axiom of" nat –choice holds for subsets"(Pow(nat))
 shows (A –is in the spectrum of" (λT. (($\bigcup$T)–is lindeloef in"T))) $\longleftrightarrow$ (A$\lesssim$nat)
 ⟨*proof*⟩

lemma lindeloef˙spec2:
 assumes $\neg$(–the axiom of" nat –choice holds for subsets"(Pow(nat)))
 shows (A –is in the spectrum of" (λT. (($\bigcup$T)–is lindeloef in"T))) $\longleftrightarrow$ Finite(A)
⟨*proof*⟩

If the axiom of countable choice on subsets of the pow of the natural numbers doesn't hold, then anti-lindeloef spaces are anti-compact.

theorem(in topology0) no˙choice˙imp˙anti˙lindeloef˙is˙anti˙comp:
 assumes $\neg$(–the axiom of" nat –choice holds for subsets"(Pow(nat))) T–is anti-lindeloef"
 shows T–is anti-compact"
⟨*proof*⟩

If the axiom of countable choice holds for subsets of the power set of the natural numbers, then there exists a topological space that is anti-lindeloef but no anti-compact.

theorem no˙choice˙imp˙anti˙lindeloef˙is˙anti˙comp:
 assumes (–the axiom of" nat –choice holds for subsets"(Pow(nat)))
 shows (–one-point compactification of"Pow(nat))–is anti-lindeloef"
⟨*proof*⟩

theorem op˙comp˙pow˙nat˙no˙anti˙comp:
 shows $\neg$((–one-point compactification of"Pow(nat))–is anti-compact")
⟨*proof*⟩

In coclusion, we reached another equivalence of this choice principle.

The axiom of countable choice holds for subsets of the power set of the natural numbers if and only if there exists a topological space which is anti-lindeloef but not anti-compact; this space can be chosen as the one-point compactification of the discrete topology on $\mathbb{N}$.

theorem acc˙pow˙nat˙equiv1:
 shows (–the axiom of" nat –choice holds for subsets"(Pow(nat))) $\longleftrightarrow$ ((–one-point compactification of"Pow(nat))–is anti-lindeloef")

⟨*proof*⟩

theorem acc˙pow˙nat˙equiv2:
shows (–the axiom of" nat –choice holds for subsets"(Pow(nat))) ⟷ (∃T. T–is a topology"
∧ (T–is anti-lindeloef") ∧ ¬(T–is anti-compact"))
⟨*proof*⟩

In the file Topology˙ZF˙properties.thy, it is proven that $\mathbb{N}$ is lindeloef if and only if the axiom of countable choice holds for subsets of $Pow(\mathbb{N})$. Now we check that, in ZF, this space is always anti-lindeloef.

theorem nat˙anti˙lindeloef:
shows Pow(nat)–is anti-lindeloef"
⟨*proof*⟩

This result is interesting because depending on the different axioms we add to ZF, it means two different things:

- Every subspace of $\mathbb{N}$ is Lindeloef.
- Only the compact subspaces of $\mathbb{N}$ are Lindeloef.

Now, we could wonder if the class of compact spaces and the class of lindeloef spaces being equal is consistent in ZF. Let's find a topological space which is lindeloef and no compact without assuming any axiom of choice or any negation of one. This will prove that the class of lindeloef spaces and the class of compact spaces cannot be equal in any model of ZF.

theorem lord˙nat:
shows (LOrdTopology nat Le)=–LeftRayX(nat,Le,n). n∈nat" ∪–nat" ∪–0"
⟨*proof*⟩

lemma countable˙lord˙nat:
shows –LeftRayX(nat,Le,n). n∈nat" ∪–nat" ∪–0"≺csucc(nat)
⟨*proof*⟩

corollary lindelof˙lord˙nat:
shows nat–is lindeloef in"(LOrdTopology nat Le)
⟨*proof*⟩

theorem not˙comp˙lord˙nat:
shows ¬(nat–is compact in"(LOrdTopology nat Le))
⟨*proof*⟩

68.4 More Separation properties

In this section we study more separation properties.

68.5 Definitions

We start with a property that has already appeared in Topology˙ZF˙1b.thy. A KC-space is a space where compact sets are closed.

definition
  IsKC (˙ –is KC″) where
  T–is KC″ ≡ ∀ A∈Pow(⋃ T). A–is compact in″T ⟶ A–is closed in″T

Another type of space is an US-space; those where sequences have at most one limit.

definition
  IsUS (˙–is US″) where
  T–is US″ ≡ ∀ N x y. (N:nat→⋃ T) ∧ NetConvTop(⟨N,Le⟩,x,T) ∧ NetConvTop(⟨N,Le⟩,y,T) ⟶ y=x

68.6 First results

The proof in Topology˙ZF˙1b.thy shows that a Hausdorff space is KC.

corollary(in topology0) T2˙imp˙KC:
  assumes T–is T_2″
  shows T–is KC″
⟨*proof*⟩

From the spectrum of compactness, it follows that any KC-space is T_1.

lemma(in topology0) KC˙imp˙T1:
  assumes T–is KC″
  shows T–is T_1″
⟨*proof*⟩

Even more, if a space is KC, then it is US. We already know that for T_2 spaces, any net or filter has at most one limit; and that this property is equivalent with T_2. The US property is much weaker because we don't know what happends with other nets that are not directed by the order on the natural numbers.

theorem(in topology0) KC˙imp˙US:
  assumes T–is KC″
  shows T–is US″
⟨*proof*⟩

US spaces are also T_1.

theorem (in topology0) US˙imp˙T1:
  assumes T–is US″
  shows T–is T_1″
⟨*proof*⟩

68.7 Counter-examples

We need to find counter-examples that prove that this properties are new ones.

We know that $T_2 \Rightarrow loc.T_2 \Rightarrow$ anti-hyperconnected $\Rightarrow T_1$ and $T_2 \Rightarrow KC \Rightarrow US \Rightarrow T_1$. The question is: What is the relation between KC or US and, $loc.T_2$ or anti-hyperconnected?

In the file Topology˙ZF˙properties˙2.thy we built a topological space which is locally-T_2 but no T_2. It happends actually that this space is not even US given the appropiate topology T.

lemma (in topology0) locT2˙not˙US˙1:
  assumes –m″∉T –m″–is closed in″T ∃N∈nat→⋃T. (⟨N,Le⟩→$_N$ m) ∧ m∉Nnat
  shows ∃N∈nat→⋃(T∪–(U-–m″)∪–⋃T″∪W. ⟨U,W⟩∈–V∈T. m∈V″×T″). (⟨N,Le⟩→$_N$ ⋃T –in″ (T∪–(U-–m″)∪–⋃T″∪W. ⟨U,W⟩∈–V∈T. m∈V″×T″))
    ∧ (⟨N,Le⟩→$_N$ m –in″ (T∪–(U-–m″)∪–⋃T″∪W. ⟨U,W⟩∈–V∈T. m∈V″×T″))
⟨*proof*⟩

corollary (in topology0) locT2˙not˙US˙2:
  assumes –m″∉T –m″–is closed in″T ∃N∈nat→⋃T. (⟨N,Le⟩→$_N$ m) ∧ m∉Nnat
  shows ¬((T∪–(U-–m″)∪–⋃T″∪W. ⟨U,W⟩∈–V∈T. m∈V″×T″)–is US″)
⟨*proof*⟩

In particular, we also know that a locally-T_2 space doesn't need to be KC; since KC⇒US. Also we know that anti-hyperconnected spaces don't need to be KC or US, since locally-T_2 ⇒anti-hyperconnected.

Let's find a KC space that is not T_2, an US space which is not KC and a T_1 space which is not US.

First, let's prove some lemmas about what relation is there between this properties under the influence of other ones. This will help us to find counter-examples.

Anti-compactness ereases the differences between several properties.

lemma (in topology0) anticompact˙KC˙equiv˙T1:
  assumes T–is anti-compact″
  shows T–is KC″⟷T–is T$_1$″
⟨*proof*⟩

Then if we find an anti-compact and T_1 but no T_2 space, there is a counter-example for $KC \Rightarrow T_2$. A counter-example for US doesn't need to be KC mustn't be anti-compact.

The cocountable topology on csucc(nat) is such a topology.

The cocountable topology on $\mathbb{N}^+$ is hyperconnected.

lemma cocountable˙in˙csucc˙nat˙HConn:

shows (CoCountable csucc(nat))–is hyperconnected″
⟨*proof*⟩

The cocountable topology on $\mathbb{N}^+$ is not anti-hyperconnected.

corollary cocountable˙in˙csucc˙nat˙notAntiHConn:
shows ¬((CoCountable csucc(nat))–is anti-″IsHConnected)
⟨*proof*⟩

The cocountable topology on $\mathbb{N}^+$ is not T_2.

theorem cocountable˙in˙csucc˙nat˙noT2:
shows ¬(CoCountable csucc(nat))–is T_2″
⟨*proof*⟩

The cocountable topology on $\mathbb{N}^+$ is T_1.

theorem cocountable˙in˙csucc˙nat˙T1:
shows (CoCountable csucc(nat))–is T_1″
⟨*proof*⟩

The cocountable topology on $\mathbb{N}^+$ is anti-compact.

theorem cocountable˙in˙csucc˙nat˙antiCompact:
shows (CoCountable csucc(nat))–is anti-compact″
⟨*proof*⟩

In conclusion, the cocountable topology defined on csucc(nat) is KC but not T_2. Also note that is KC but not anti-hyperconnected, hence KC or US spaces need not to be sober.

The cofinite topology on the natural numbers is T_1, but not US.

theorem cofinite˙not˙US:
shows ¬((CoFinite nat)–is US″)
⟨*proof*⟩

To end, we need a space which is US but no KC. This example comes from the one point compactification of a T_2, anti-compact and non discrete space. This T_2, anti-compact and non discrete space comes from a construction over the cardinal $\mathbb{N}^+$ or csucc(nat).

theorem extension˙pow˙top:
shows (Pow(csucc(nat)) ∪ —csucc(nat)″∪S. S∈(CoCountable csucc(nat))-–0″″)–is a topology″
⟨*proof*⟩

This topology is defined over $\mathbb{N}^+ \cup \{\mathbb{N}^+\}$ or csucc(nat)∪–csucc(nat)″.

lemma extension˙pow˙union:
shows ⋃(Pow(csucc(nat)) ∪ —csucc(nat)″∪S. S∈(CoCountable csucc(nat))-–0″″)=csucc(nat)∪–csucc(nat)″
⟨*proof*⟩

This topology has a discrete open subspace.

lemma extension˙pow˙subspace:
shows (Pow(csucc(nat)) ∪ —csucc(nat)″∪S. S∈(CoCountable csucc(nat))-–0″″)–restricted to″csucc(nat)=Pow(csucc(nat))
and csucc(nat)∈(Pow(csucc(nat)) ∪ —csucc(nat)″∪S. S∈(CoCountable csucc(nat))-–0″″)
⟨*proof*⟩

This topology is Hausdorff.

theorem extension˙pow˙T2:
shows (Pow(csucc(nat)) ∪ —csucc(nat)″∪S. S∈(CoCountable csucc(nat))-–0″″)–is T_2″
⟨*proof*⟩

The topology we built is not discrete; i.e., not every set is open.

theorem extension˙pow˙notDiscrete:
shows –csucc(nat)″∉(Pow(csucc(nat)) ∪ —csucc(nat)″∪S. S∈(CoCountable csucc(nat))-–0″″)
⟨*proof*⟩

The topology we built is anti-compact.

theorem extension˙pow˙antiCompact:
shows (Pow(csucc(nat)) ∪ —csucc(nat)″∪S. S∈(CoCountable csucc(nat))-–0″″)–is anti-compact″
⟨*proof*⟩

If a topological space is KC, then its one-point compactification is US.

theorem (in topology0) KC˙imp˙OP˙comp˙is˙US:
assumes T–is KC″
shows (–one-point compactification of″T)–is US″
⟨*proof*⟩

In the one-point compactification of an anti-compact space, ever subspace that contains the infinite point is compact.

theorem (in topology0) anti˙comp˙imp˙OP˙inf˙comp:
assumes T–is anti-compact″ A⊆⋃(–one-point compactification of″T) ⋃T∈A
shows A–is compact in″(–one-point compactification of″T)
⟨*proof*⟩

As a last result in this section, the one-point compactification of our topology is not a KC space.

theorem extension˙pow˙OP˙not˙KC:
shows ¬((–one-point compactification of″(Pow(csucc(nat)) ∪ —csucc(nat)″∪S. S∈(CoCountable csucc(nat))-–0″″))–is KC″)
⟨*proof*⟩

In conclusion, $US \nRightarrow KC$.

68.8 Other types of properties

In this section we will define new properties that aren't defined as anti-properties and that are not separation axioms. In some cases we will consider their anti-properties.

68.9 Definitions

A space is called perfect if it has no isolated points. This definition may vary in the literature to similar, but not equivalent definitions.

definition
IsPerf (˙ –is perfect˝) where
T–is perfect˝ ≡ ∀x∈⋃T. –x˝∉T

An anti-perfect space is called scattered.

definition
IsScatt (˙ –is scattered˝) where
T–is scattered˝ ≡ T–is anti-˝IsPerf

A topological space with two disjoint dense subspaces is called resolvable.

definition
IsRes (˙ –is resolvable˝) where
T–is resolvable˝ ≡ ∃U∈Pow(⋃T). ∃V∈Pow(⋃T). Closure(U,T)=⋃T ∧ Closure(V,T)=⋃T ∧ U∩V=0

A topological space where every dense subset is open is called submaximal.

definition
IsSubMax (˙ –is submaximal˝) where
T–is submaximal˝ ≡ ∀U∈Pow(⋃T). Closure(U,T)=⋃T ⟶ U∈T

A subset of a topological space is nowhere-dense if the interior of its closure is empty.

definition
IsNowhereDense (˙ –is nowhere dense in˝ ˙) where
A–is nowhere dense in˝T ≡ A⊆⋃T ∧ Interior(Closure(A,T),T)=0

A topological space is then a Luzin space if every nowhere-dense subset is countable.

definition
IsLuzin (˙ –is luzin˝) where
T–is luzin˝ ≡ ∀A∈Pow(⋃T). (A–is nowhere dense in˝T) ⟶ A≲nat

An also useful property is local-connexion.

definition
IsLocConn (˙–is locally-connected˝) where

T–is locally-connected″ ≡ T–is locally″(λT. λB. ((T–restricted to″B)–is connected″))

An SI-space is an anti-resolvable perfect space.

definition
IsAntiRes (˙–is anti-resolvable″) where
T–is anti-resolvable″ ≡ T–is anti-″IsRes

definition
IsSI (˙–is Strongly Irresolvable″) where
T–is Strongly Irresolvable″ ≡ (T–is anti-resolvable″) ∧ (T–is perfect″)

68.10 First examples

Firstly, we need to compute the spectrum of the being perfect.

lemma spectrum˙perfect:
shows (A–is in the spectrum of″IsPerf) ⟷ A=0
⟨*proof*⟩

The discrete space is clearly scattered:

lemma pow˙is˙scattered:
shows Pow(A)–is scattered″
⟨*proof*⟩

The trivial topology is perfect, if it is defined over a set with more than one point.

lemma trivial˙is˙perfect:
assumes ∃x y. x∈X ∧ y∈X ∧ x≠y
shows –0,X″–is perfect″
⟨*proof*⟩

The trivial topology is resolvable, if it is defined over a set with more than one point.

lemma trivial˙is˙resolvable:
assumes ∃x y. x∈X ∧ y∈X ∧ x≠y
shows –0,X″–is resolvable″
⟨*proof*⟩

The spectrum of Luzin spaces is the class of countable sets, so there are lots of examples of Luzin spaces.

lemma spectrum˙Luzin:
shows (A–is in the spectrum of″IsLuzin) ⟷ A≲nat
⟨*proof*⟩

68.11 Structural results

Every resolvable space is also perfect.

theorem (in topology0) resolvable˙imp˙perfect:
  assumes T–is resolvable″
  shows T–is perfect″
⟨*proof*⟩

The spectrum of being resolvable follows:

corollary spectrum˙resolvable:
  shows (A–is in the spectrum of″IsRes) ⟷ A=0
⟨*proof*⟩

The cofinite space over $\mathbb{N}$ is a T_1, perfect and luzin space.

theorem cofinite˙nat˙perfect:
  shows (CoFinite nat)–is perfect″
⟨*proof*⟩

theorem cofinite˙nat˙luzin:
  shows (CoFinite nat)–is luzin″
⟨*proof*⟩

The cocountable topology on $\mathbb{N}^+$ or csucc(nat) is also T_1, perfect and luzin; but defined on a set not in the spectrum.

theorem cocountable˙csucc˙nat˙perfect:
  shows (CoCountable csucc(nat))–is perfect″
⟨*proof*⟩

theorem cocountable˙csucc˙nat˙luzin:
  shows (CoCountable csucc(nat))–is luzin″
⟨*proof*⟩

The existence of T_2, uncountable, perfect and luzin spaces is unprovable in ZFC. It is related to the CH and Martin's axiom.

end

69 Metric spaces

theory MetricSpace˙ZF imports Topology˙ZF˙1 OrderedGroup˙ZF Lattice˙ZF
begin

A metric space is a set on which a distance between points is defined as a function $d : X \times X \to [0, \infty)$. With this definition each metric space is a topological space which is paracompact and Hausdorff (T_2), hence normal (in fact even perfectly normal).

69.1 Pseudometric - definition and basic properties

A metric on X is usually defined as a function $d : X \times X \to [0, \infty)$ that satisfies the conditions $d(x, x) = 0$, $d(x, y) = 0 \Rightarrow x = y$ (identity of indiscernibles), $d(x, y) = d(y, x)$ (symmetry) and $d(x, y) \leq d(x, z) + d(z, y)$

(triangle inequality) for all $x, y \in X$. Here we are going to be a bit more general and define metric and pseudo-metric as a function valued in an ordered group.

First we define a pseudo-metric, which has the axioms of a metric, but without the second part of the identity of indiscernibles. In our definition IsApseudoMetric is a predicate on five sets: the function d, the set X on which the metric is defined, the group carrier G, the group operation A and the order r on G.

definition
  IsApseudoMetric(d,X,G,A,r) ≡ d:X×X → Nonnegative(G,A,r)
    ∧ (∀x∈X. d⟨x,x⟩ = TheNeutralElement(G,A))
    ∧ (∀x∈X.∀y∈X. d⟨x,y⟩ = d⟨y,x⟩)
    ∧ (∀x∈X.∀y∈X.∀z∈X. ⟨d⟨x,z⟩, A⟨d⟨x,y⟩,d⟨y,z⟩⟩⟩ ∈ r)

We add the full axiom of identity of indiscernibles to the definition of a pseudometric to get the definition of metric.

definition
  IsAmetric(d,X,G,A,r) ≡
  IsApseudoMetric(d,X,G,A,r) ∧ (∀x∈X.∀y∈X. d⟨x,y⟩ = TheNeutralElement(G,A) ⟶ x=y)

A disk is defined as set of points located less than the radius from the center.

definition Disk(X,d,r,c,R) ≡ –x∈X. ⟨d⟨c,x⟩,R⟩ ∈ StrictVersion(r)″

Next we define notation for metric spaces. We will use additive notation for the group operation but we do not assume that the group is abelian. Since for many theorems it is sufficient to assume the pseudometric axioms we will assume in this context that the sets d, X, G, A, r form a pseudometric raher than a metric.

locale pmetric˙space =
  fixes d and X and G and A and r

  assumes ordGroupAssum: IsAnOrdGroup(G,A,r)

  assumes pmetricAssum: IsApseudoMetric(d,X,G,A,r)

  fixes zero (**0**)
  defines zero˙def [simp]: **0** ≡ TheNeutralElement(G,A)

  fixes grop (infixl + 80)
  defines grop˙def [simp]: x+y ≡ A⟨x,y⟩

  fixes grinv (- ˙ 89)
  defines grinv˙def [simp]: (-x) ≡ GroupInv(G,A)(x)

  fixes lesseq (infix ≤ 68)

defines lesseq˙def [simp]: x $\leq$ y $\equiv$ $\langle$ x,y$\rangle$ $\in$ r

fixes sless (infix ¡ 68)
defines sless˙def [simp]: x ¡ y $\equiv$ x$\leq$y $\wedge$ x$\neq$y

fixes nonnegative (G^+)
defines nonnegative˙def [simp]: G^+ $\equiv$ Nonnegative(G,A,r)

fixes positive (G_+)
defines positive˙def [simp]: G_+ $\equiv$ PositiveSet(G,A,r)

fixes setinv (- ˙ 72)
defines setninv˙def [simp]: -C $\equiv$ GroupInv(G,A)(C)

fixes abs
defines abs˙def [simp]: abs(x) $\equiv$ AbsoluteValue(G,A,r)(x)

fixes oddext (˙ °)
defines oddext˙def [simp]: f° $\equiv$ OddExtension(G,A,r,f)

fixes disk
defines disk˙def [simp]: disk(c,R) $\equiv$ Disk(X,d,r,c,R)

The theorems proven the in the group3 locale are valid in the pmetric˙space locale.

sublocale
pmetric˙space ¡ group3 G A r zero grop grinv lesseq sless nonnegative positive setinv abs oddext
$\langle$*proof*$\rangle$

The theorems proven the in the group0 locale are valid in the pmetric˙space locale.

sublocale pmetric˙space ¡ group0 G A zero grop grinv
$\langle$*proof*$\rangle$

The next lemma shows the definition of the pseudometric in the notation used in the metric˙space context.

lemma (in pmetric˙space) pmetric˙properties: shows
d: X$\times$X $\rightarrow$ G^+
$\forall$x$\in$X. d$\langle$x,x$\rangle$ = **0**
$\forall$x$\in$X.$\forall$y$\in$X. d$\langle$x,y$\rangle$ = d$\langle$y,x$\rangle$
$\forall$x$\in$X.$\forall$y$\in$X.$\forall$z$\in$X. d$\langle$x,z$\rangle$ $\leq$ d$\langle$x,y$\rangle$ + d$\langle$y,z$\rangle$
$\langle$*proof*$\rangle$

The definition of the disk in the notation used in the pmetric˙space context:

lemma (in pmetric˙space) disk˙definition: shows disk(c,R) = –x$\in$X. d$\langle$c,x$\rangle$ ¡ R˝
$\langle$*proof*$\rangle$

If the radius is positive then the center is in disk.

lemma (in pmetric˙space) center˙in˙disk: assumes c∈X and R∈G_+ shows c ∈ disk(c,R)
⟨*proof*⟩

A technical lemma that allows us to shorten some proofs:

lemma (in pmetric˙space) radius˙in˙group: assumes c∈X and x ∈ disk(c,R)
shows R∈G **0**¡R R∈G_+ ((-d⟨c,x⟩) + R) ∈ G_+
⟨*proof*⟩

If a point x is inside a disk B and $m \leq R - d(c, x)$ then the disk centered at the point x and with radius m is contained in the disk B.

lemma (in pmetric˙space) disk˙in˙disk:
assumes c∈X and x ∈ disk(c,R) and m ≤ (-d⟨c,x⟩) + R
shows disk(x,m) ⊆ disk(c,R)
⟨*proof*⟩

If we assume that the order on the group makes the positive set a meet semi-lattice (i.e. every two-element subset of G_+ has a greatest lower bound) then the collection of disks centered at points of the space and with radii in the positive set of the group satisfies the base condition. The meet semi-lattice assumption can be weakened to "each two-element subset of G_+ has a lower bound in G_+", but we don't do that here.

lemma (in pmetric˙space) disks˙form˙base:
assumes IsMeetSemilattice(G_+,r ∩ G_+×G_+)
defines B ≡ ⋃c∈X. –disk(c,R). R∈G_+″
shows B –satisfies the base condition″
⟨*proof*⟩

Unions of disks form a topology, hence (pseudo)metric spaces are topological spaces. We have to add the assumption that the positive set is not empty. This is necessary to show that we can cover the space with disks and it does not look like it follows from anything we have assumed so far.

theorem (in pmetric˙space) pmetric˙is˙top:
assumes IsMeetSemilattice(G_+,r ∩ G_+×G_+) G_+≠0
defines B ≡ ⋃c∈X. –disk(c,R). R∈G_+″
defines T ≡ –⋃A. A ∈ Pow(B)″
shows T –is a topology″ B –is a base for″ T ⋃T = X
⟨*proof*⟩

end

70 Uniform spaces

theory UniformSpace˙ZF imports Topology˙ZF˙4a
begin

This theory defines uniform spaces and proves their basic properties.

70.1 Definition and motivation

Just like a topological space constitutes the minimal setting in which one can speak of continuous functions, the notion of uniform spaces (commonly attributed to Andr Weil) captures the minimal setting in which one can speak of uniformly continuous functions. In some sense this is a generalization of the notion of metric (or metrizable) spaces and topological groups.

There are several definitions of uniform spaces. The fact that these definitions are equivalent is far from obvious (some people call such phenomenon cryptomorphism). We will use the definition of the uniform structure (or "uniformity") based on entourages. This was the original definition by Weil and it seems to be the most commonly used. A uniformity consists of entourages that are binary relations between points of space X that satisfy a certain collection of conditions, specified below.

definition
  IsUniformity (˙ –is a uniformity on″ ˙ 90) where
    Φ –is a uniformity on″ X ≡(Φ –is a filter on″ (X×X))
    ∧ (∀U∈Φ. id(X) ⊆ U ∧ (∃V∈Φ. V O V ⊆ U) ∧ converse(U) ∈ Φ)

If Φ is a uniformity on X, then the every element V of Φ is a certain relation on X (a subset of $X \times X$) and is called an "entourage". For an $x \in X$ we call $V\{x\}$ a neighborhood of x. The first useful fact we will show is that neighborhoods are non-empty.

lemma neigh˙not˙empty:
  assumes Φ –is a uniformity on″ X V∈Φ and x∈X
  shows V–x″ ≠ 0 and x ∈ V–x″
⟨*proof*⟩

Uniformity Φ defines a natural topology on its space X via the neighborhood system that assigns the collection $\{V(\{x\}) : V \in \Phi\}$ to every point $x \in X$. In the next lemma we show that if we define a function this way the values of that function are what they should be. This is only a technical fact which is useful to shorten the remaining proofs, usually treated as obvious in standard mathematics.

lemma neigh˙filt˙fun:
  assumes Φ –is a uniformity on″ X
  defines $\mathcal{M}$ ≡ –⟨x,–V–x″.V∈Φ″⟩.x∈X″
  shows $\mathcal{M}$:X→Pow(Pow(X)) and ∀x∈X. $\mathcal{M}$(x) = –V–x″.V∈Φ″
⟨*proof*⟩

In the next lemma we show that the collection defined in lemma neigh˙filt˙fun is a filter on X. The proof is kind of long, but it just checks that all filter conditions hold.

lemma filter˙from˙uniformity:
  assumes Φ –is a uniformity on″ X and x∈X

defines $\mathcal{M} \equiv$ –⟨x,–V–x″.V∈Φ″⟩.x∈X″
shows $\mathcal{M}$(x) –is a filter on″ X
⟨*proof*⟩

The function defined in the premises of lemma neigh˙filt˙fun (or filter˙from˙uniformity) is a neighborhood system. The proof uses the existence of the "half-the-size" neighborhood condition (∃ V∈Φ. V O V ⊆ U) of the uniformity definition, but not the converse(U) ∈ Φ part.

theorem neigh˙from˙uniformity:
assumes Φ –is a uniformity on″ X
shows –⟨x,–V–x″.V∈Φ″⟩.x∈X″ –is a neighborhood system on″ X
⟨*proof*⟩

When we have a uniformity Φ on X we can define a topology on X in a (relatively) natural way. We will call that topology the UniformTopology(Φ). The definition may be a bit cryptic but it just combines the construction of a neighborhood system from uniformity as in the assumptions of lemma filter˙from˙uniformity and the construction of topology from a neighborhood system from theorem topology˙from˙neighs. We could probably reformulate the definition to skip the X parameter because if Φ is a uniformity on X then X can be recovered from (is determined by) Φ.

definition
UniformTopology(Φ,X) ≡ –U ∈ Pow(X). ∀x∈U. U ∈ –⟨t,–V–t″.V∈Φ″⟩.t∈X″(x)″

The collection of sets constructed in the UniformTopology definition is indeed a topology on X.

theorem uniform˙top˙is˙top:
assumes Φ –is a uniformity on″ X
shows
UniformTopology(Φ,X) –is a topology″ and ⋃ UniformTopology(Φ,X) = X
⟨*proof*⟩

end

71 More on uniform spaces

theory UniformSpace˙ZF˙1 imports func˙ZF˙1 UniformSpace˙ZF Topology˙ZF˙2
begin

This theory defines the maps to study in uniform spaces and proves their basic properties.

71.1 Uniformly continuous functions

Just as the the most general setting for continuity of functions is that of topological spaces, uniform spaces are the most general setting for the study of uniform continuity.

A map between 2 uniformities is uniformly continuous if it preserves the entourages:

definition
  IsUniformlyCont (˙ –is uniformly continuous between″ ˙ –and″ ˙ 90) where
  f:X-¿Y ==¿ Φ –is a uniformity on″X ==¿ Γ –is a uniformity on″Y ==¿
 f –is uniformly continuous between″ Φ –and″ Γ ≡ ∀ V∈Γ. (ProdFunction(f,f)-V)∈Φ

Any uniformly continuous function is continuous when considering the topologies on the uniformities.

lemma uniformly˙cont˙is˙cont:
  assumes f:X-¿Y Φ –is a uniformity on″X Γ –is a uniformity on″Y
    f –is uniformly continuous between″ Φ –and″ Γ
  shows IsContinuous(UniformTopology(Φ,X),UniformTopology(Γ,Y),f)
⟨*proof*⟩

end

72 Topological groups - introduction

theory TopologicalGroup˙ZF imports Topology˙ZF˙3 Group˙ZF˙1 Semigroup˙ZF

begin

This theory is about the first subject of algebraic topology: topological groups.

72.1 Topological group: definition and notation

Topological group is a group that is a topological space at the same time. This means that a topological group is a triple of sets, say (G, f, T) such that T is a topology on G, f is a group operation on G and both f and the operation of taking inverse in G are continuous. Since IsarMathLib defines topology without using the carrier, (see Topology˙ZF), in our setup we just use $\bigcup T$ instead of G and say that the pair of sets $(\bigcup T, f)$ is a group. This way our definition of being a topological group is a statement about two sets: the topology T and the group operation f on $G = \bigcup T$. Since the domain of the group operation is $G \times G$, the pair of topologies in which f is supposed to be continuous is T and the product topology on $G \times G$ (which we will call τ below).

This way we arrive at the following definition of a predicate that states that pair of sets is a topological group.

definition
  IsAtopologicalGroup(T,f) ≡ (T –is a topology″) ∧ IsAgroup(⋃T,f) ∧
  IsContinuous(ProductTopology(T,T),T,f) ∧
  IsContinuous(T,T,GroupInv(⋃T,f))

We will inherit notation from the topology0 locale. That locale assumes that T is a topology. For convenience we will denote $G = \bigcup T$ and τ to be the product topology on $G \times G$. To that we add some notation specific to groups. We will use additive notation for the group operation, even though we don't assume that the group is abelian. The notation $g + A$ will mean the left translation of the set A by element g, i.e. $g + A = \{g + a | a \in A\}$. The group operation G induces a natural operation on the subsets of G defined as $\langle A, B\rangle \mapsto \{x + y | x \in A, y \in B\}$. Such operation has been considered in func˙ZF and called f "lifted to subsets of" G. We will denote the value of such operation on sets A, B as $A + B$. The set of neigboorhoods of zero (denoted $\mathcal{N}_0$) is the collection of (not necessarily open) sets whose interior contains the neutral element of the group.

locale topgroup = topology0 +

fixes G
defines G˙def [simp]: G ≡ $\bigcup$T

fixes prodtop (τ)
defines prodtop˙def [simp]: τ ≡ ProductTopology(T,T)

fixes f

assumes Ggroup: IsAgroup(G,f)

assumes fcon: IsContinuous(τ,T,f)

assumes inv˙cont: IsContinuous(T,T,GroupInv(G,f))

fixes grop (infixl + 90)
defines grop˙def [simp]: x+y ≡ f⟨x,y⟩

fixes grinv (- ˙ 89)
defines grinv˙def [simp]: (-x) ≡ GroupInv(G,f)(x)

fixes grsub (infixl - 90)
defines grsub˙def [simp]: x-y ≡ x+(-y)

fixes setinv (- ˙ 72)
defines setninv˙def [simp]: -A ≡ GroupInv(G,f)(A)

fixes ltrans (infix + 73)
defines ltrans˙def [simp]: x + A ≡ LeftTranslation(G,f,x)(A)

fixes rtrans (infix + 73)
defines rtrans˙def [simp]: A + x ≡ RightTranslation(G,f,x)(A)

fixes setadd (infixl + 71)

defines setadd˙def [simp]: A+B ≡ (f –lifted to subsets of″ G)⟨A,B⟩

fixes gzero (**0**)
defines gzero˙def [simp]: **0** ≡ TheNeutralElement(G,f)

fixes zerohoods ($\mathcal{N}_0$)
defines zerohoods˙def [simp]: $\mathcal{N}_0$ ≡ –A ∈ Pow(G). **0** ∈ int(A)″

fixes listsum ($\sum$ ˙ 70)
defines listsum˙def[simp]: $\sum$k ≡ Fold1(f,k)

The first lemma states that we indeeed talk about topological group in the context of topgroup locale.

lemma (in topgroup) topGroup: shows IsAtopologicalGroup(T,f)
⟨*proof*⟩

If a pair of sets (T, f) forms a topological group, then all theorems proven in the topgroup context are valid as applied to (T, f).

lemma topGroupLocale: assumes IsAtopologicalGroup(T,f)
shows topgroup(T,f)
⟨*proof*⟩

We can use the group0 locale in the context of topgroup.

lemma (in topgroup) group0˙valid˙in˙tgroup: shows group0(G,f)
⟨*proof*⟩

We can use the group0 locale in the context of topgroup.

sublocale topgroup ¡ group0 G f gzero grop grinv
⟨*proof*⟩

We can use semigr0 locale in the context of topgroup.

lemma (in topgroup) semigr0˙valid˙in˙tgroup: shows semigr0(G,f)
⟨*proof*⟩

We can use the prod˙top˙spaces0 locale in the context of topgroup.

lemma (in topgroup) prod˙top˙spaces0˙valid: shows prod˙top˙spaces0(T,T,T)
⟨*proof*⟩

Negative of a group element is in group.

lemma (in topgroup) neg˙in˙tgroup: assumes g∈G shows (-g) ∈ G
⟨*proof*⟩

Sum of two group elements is in the group.

lemma (in topgroup) group˙op˙closed˙add: assumes x_1 ∈ G x_2 ∈ G
shows x_1+x_2 ∈ G
⟨*proof*⟩

Zero is in the group.

lemma (in topgroup) zero_in_tgroup: shows $\mathbf{0} \in G$
  ⟨*proof*⟩

Another lemma about canceling with two group elements written in additive notation

lemma (in topgroup) inv_cancel_two_add:
  assumes $x_1 \in G$ $x_2 \in G$
  shows
    $x_1+(-x_2)+x_2 = x_1$
    $x_1+x_2+(-x_2) = x_1$
    $(-x_1)+(x_1+x_2) = x_2$
    $x_1+((-x_1)+x_2) = x_2$
  ⟨*proof*⟩

Useful identities proven in the Group_ZF theory, rewritten here in additive notation. Note since the group operation notation is left associative we don't really need the first set of parentheses in some cases.

lemma (in topgroup) cancel_middle_add: assumes $x_1 \in G$ $x_2 \in G$ $x_3 \in G$
  shows
    $(x_1+(-x_2))+(x_2+(-x_3)) = x_1+ (-x_3)$
    $((-x_1)+x_2)+((-x_2)+x_3) = (-x_1)+ x_3$
    $(- (x_1+x_2)) + (x_1+x_3) = (-x_2)+x_3$
    $(x_1+x_2) + (-(x_3+x_2)) =x_1+ (-x_3)$
    $(-x_1) + (x_1+x_2+x_3) + (-x_3) = x_2$
⟨*proof*⟩

We can cancel an element on the right from both sides of an equation.

lemma (in topgroup) cancel_right_add:
  assumes $x_1 \in G$ $x_2 \in G$ $x_3 \in G$ $x_1+x_2 = x_3+x_2$
  shows $x_1 = x_3$
  ⟨*proof*⟩

We can cancel an element on the left from both sides of an equation.

lemma (in topgroup) cancel_left_add:
  assumes $x_1 \in G$ $x_2 \in G$ $x_3 \in G$ $x_1+x_2 = x_1+x_3$
  shows $x_2 = x_3$
  ⟨*proof*⟩

We can put an element on the other side of an equation.

lemma (in topgroup) put_on_the_other_side:
  assumes $x_1 \in G$ $x_2 \in G$ $x_3 = x_1+x_2$
  shows $x_3+(-x_2) = x_1$ and $(-x_1)+x_3 = x_2$
  ⟨*proof*⟩

A simple equation from lemma simple_equation0 in Group_ZF in additive notation

lemma (in topgroup) simple˙equation0˙add:
  assumes $x_1 \in G$ $x_2 \in G$ $x_3 \in G$ $x_1+(-x_2) = (-x_3)$
  shows $x_3 = x_2 + (-x_1)$
  ⟨*proof*⟩

A simple equation from lemma simple˙equation1 in Group˙ZF in additive notation

lemma (in topgroup) simple˙equation1˙add:
  assumes $x_1 \in G$ $x_2 \in G$ $x_3 \in G$ $(-x_1)+x_2 = (-x_3)$
  shows $x_3 = (-x_2) + x_1$
  ⟨*proof*⟩

The set comprehension form of negative of a set. The proof uses the ginv˙image lemma from Group˙ZF theory which states the same thing in multiplicative notation.

lemma (in topgroup) ginv˙image˙add: assumes V⊆G
  shows (-V)⊆G and (-V) = –-x. x ∈ V″
  ⟨*proof*⟩

The additive notation version of ginv˙image˙el lemma from Group˙ZF theory

lemma (in topgroup) ginv˙image˙el˙add: assumes V⊆G x ∈ (-V)
  shows (-x) ∈ V
  ⟨*proof*⟩

Of course the product topology is a topology (on $G \times G$).

lemma (in topgroup) prod˙top˙on˙G:
  shows τ –is a topology″ and $\bigcup \tau$ = G×G
  ⟨*proof*⟩

Let's recall that f is a binary operation on G in this context.

lemma (in topgroup) topgroup˙f˙binop: shows f : G×G → G
  ⟨*proof*⟩

A subgroup of a topological group is a topological group with relative topology and restricted operation. Relative topology is the same as T –restricted to″ H which is defined to be $\{V \cap H : V \in T\}$ in ZF1 theory.

lemma (in topgroup) top˙subgroup: assumes A1: IsAsubgroup(H,f)
  shows IsAtopologicalGroup(T –restricted to″ H,restrict(f,H×H))
⟨*proof*⟩

72.2 Interval arithmetic, translations and inverse of set

In this section we list some properties of operations of translating a set and reflecting it around the neutral element of the group. Many of the results are proven in other theories, here we just collect them and rewrite in notation specific to the topgroup context.

Different ways of looking at adding sets.

lemma (in topgroup) interval˙add: assumes A⊆G B⊆G shows
  A+B ⊆ G
  A+B = f(A×B)
  A+B = (⋃x∈A. x+B)
  A+B = –x+y. ⟨x,y⟩ ∈ A×B″
⟨*proof*⟩

If the neutral element is in a set, then it is in the sum of the sets.

lemma (in topgroup) interval˙add˙zero: assumes A⊆G **0**∈A
  shows **0** ∈ A+A
⟨*proof*⟩

Some lemmas from Group˙ZF˙1 about images of set by translations written in additive notation

lemma (in topgroup) lrtrans˙image: assumes V⊆G x∈G
  shows
    x+V = –x+v. v∈V″
    V+x = –v+x. v∈V″
  ⟨*proof*⟩

Right and left translations of a set are subsets of the group. This is of course typically applied to the subsets of the group, but formally we don't need to assume that.

lemma (in topgroup) lrtrans˙in˙group˙add: assumes x∈G
  shows x+V ⊆ G and V+x ⊆G
  ⟨*proof*⟩

A corollary from interval˙add

corollary (in topgroup) elements˙in˙set˙sum: assumes A⊆G B⊆G
  t ∈ A+B shows ∃s∈A. ∃q∈B. t=s+q
  ⟨*proof*⟩

A corollary from lrtrans˙image

corollary (in topgroup) elements˙in˙ltrans:
  assumes B⊆G g∈G t ∈ g+B
  shows ∃q∈B. t=g+q
  ⟨*proof*⟩

Another corollary of lrtrans˙image

corollary (in topgroup) elements˙in˙rtrans:
  assumes B⊆G g∈G t ∈ B+g shows ∃q∈B. t=q+g
  ⟨*proof*⟩

Another corollary from interval˙add

corollary (in topgroup) elements˙in˙set˙sum˙inv:
  assumes A⊆G B⊆G t=s+q s∈A q∈B

shows t ∈ A+B
⟨*proof*⟩

Another corollary of lrtrans˙image

corollary (in topgroup) elements˙in˙ltrans˙inv: assumes B⊆G g∈G q∈B t=g+q
shows t ∈ g+B
⟨*proof*⟩

Another corollary of rtrans˙image˙add

lemma (in topgroup) elements˙in˙rtrans˙inv:
assumes B⊆G g∈G q∈B t=q+g
shows t ∈ B+g
⟨*proof*⟩

Right and left translations are continuous.

lemma (in topgroup) trans˙cont: assumes g∈G shows
IsContinuous(T,T,RightTranslation(G,f,g)) and
IsContinuous(T,T,LeftTranslation(G,f,g))
⟨*proof*⟩

Left and right translations of an open set are open.

lemma (in topgroup) open˙tr˙open: assumes g∈G and V∈T
shows g+V ∈ T and V+g ∈ T
⟨*proof*⟩

Right and left translations are homeomorphisms.

lemma (in topgroup) tr˙homeo: assumes g∈G shows
IsAhomeomorphism(T,T,RightTranslation(G,f,g)) and
IsAhomeomorphism(T,T,LeftTranslation(G,f,g))
⟨*proof*⟩

Left translations preserve interior.

lemma (in topgroup) ltrans˙interior: assumes A1: g∈G and A2: A⊆G
shows g + int(A) = int(g+A)
⟨*proof*⟩

Right translations preserve interior.

lemma (in topgroup) rtrans˙interior: assumes A1: g∈G and A2: A⊆G
shows int(A) + g = int(A+g)
⟨*proof*⟩

Translating by an inverse and then by an element cancels out.

lemma (in topgroup) trans˙inverse˙elem: assumes g∈G and A⊆G
shows g+((-g)+A) = A
⟨*proof*⟩

Inverse of an open set is open.

lemma (in topgroup) open˙inv˙open: assumes V∈T shows (-V) ∈ T
⟨*proof*⟩

Inverse is a homeomorphism.

lemma (in topgroup) inv˙homeo: shows IsAhomeomorphism(T,T,GroupInv(G,f))
⟨*proof*⟩

Taking negative preserves interior.

lemma (in topgroup) int˙inv˙inv˙int: assumes A ⊆ G
shows int(-A) = -(int(A))
⟨*proof*⟩

72.3 Neighborhoods of zero

Zero neighborhoods are (not necessarily open) sets whose interior contains the neutral element of the group. In the topgroup locale the collection of neighboorhoods of zero is denoted $\mathcal{N}_0$.

The whole space is a neighborhood of zero.

lemma (in topgroup) zneigh˙not˙empty: shows G ∈ $\mathcal{N}_0$
⟨*proof*⟩

Any element that belongs to a subset of the group belongs to that subset with the interior of a neighborhood of zero added.

lemma (in topgroup) elem˙in˙int˙sad: assumes A⊆G g∈A H ∈ $\mathcal{N}_0$
shows g ∈ A+int(H)
⟨*proof*⟩

Any element belongs to the interior of any neighboorhood of zero left translated by that element.

lemma (in topgroup) elem˙in˙int˙ltrans:
assumes g∈G and H ∈ $\mathcal{N}_0$
shows g ∈ int(g+H) and g ∈ int(g+H) + int(H)
⟨*proof*⟩

Any element belongs to the interior of any neighboorhood of zero right translated by that element.

lemma (in topgroup) elem˙in˙int˙rtrans:
assumes A1: g∈G and A2: H ∈ $\mathcal{N}_0$
shows g ∈ int(H+g) and g ∈ int(H+g) + int(H)
⟨*proof*⟩

Negative of a neighborhood of zero is a neighborhood of zero.

lemma (in topgroup) neg˙neigh˙neigh: assumes H ∈ $\mathcal{N}_0$
shows (-H) ∈ $\mathcal{N}_0$
⟨*proof*⟩

Left translating an open set by a negative of a point that belongs to it makes it a neighboorhood of zero.

lemma (in topgroup) open˙trans˙neigh: assumes A1: U∈T and g∈U
  shows (-g)+U ∈ $\mathcal{N}_0$
⟨*proof*⟩

Right translating an open set by a negative of a point that belongs to it makes it a neighboorhood of zero.

lemma (in topgroup) open˙trans˙neigh˙2: assumes A1: U∈T and g∈U
  shows U+(-g) ∈ $\mathcal{N}_0$
⟨*proof*⟩

Right and left translating an neighboorhood of zero by a point and its negative makes it back a neighboorhood of zero.

lemma (in topgroup) lrtrans˙neigh: assumes W∈$\mathcal{N}_0$ and x∈G
  shows x+(W+(-x)) ∈ $\mathcal{N}_0$ and (x+W)+(-x) ∈ $\mathcal{N}_0$
⟨*proof*⟩

If A is a subset of B translated by $-x$ then its translation by x is a subset of B.

lemma (in topgroup) trans˙subset:
  assumes A ⊆ ((-x)+B)x∈G B⊆G
  shows x+A ⊆ B
⟨*proof*⟩

Every neighborhood of zero has a symmetric subset that is a neighborhood of zero.

theorem (in topgroup) exists˙sym˙zerohood:
  assumes U∈$\mathcal{N}_0$
  shows ∃V∈$\mathcal{N}_0$. (V⊆U ∧ (-V)=V)
⟨*proof*⟩

We can say even more than in exists˙sym˙zerohood: every neighborhood of zero U has a symmetric subset that is a neighborhood of zero and its set double is contained in U.

theorem (in topgroup) exists˙procls˙zerohood:
  assumes U∈$\mathcal{N}_0$
  shows ∃V∈$\mathcal{N}_0$. (V⊆U∧ (V+V)⊆U ∧ (-V)=V)
⟨*proof*⟩

72.4 Closure in topological groups

This section is devoted to a characterization of closure in topological groups.

Closure of a set is contained in the sum of the set and any neighboorhood of zero.

lemma (in topgroup) cl˙contains˙zneigh:
  assumes A1: A⊆G and A2: H ∈ $\mathcal{N}_0$
  shows cl(A) ⊆ A+H
⟨*proof*⟩

The next theorem provides a characterization of closure in topological groups in terms of neighborhoods of zero.

theorem (in topgroup) cl˙topgroup:
  assumes A⊆G shows cl(A) = (⋂H∈$\mathcal{N}_0$. A+H)
⟨*proof*⟩

72.5 Sums of sequences of elements and subsets

In this section we consider properties of the function $G^n \to G, x = (x_0, x_1, ..., x_{n-1}) \mapsto \sum_{i=0}^{n-1} x_i$. We will model the cartesian product G^n by the space of sequences $n \to G$, where $n = \{0, 1, ..., n-1\}$ is a natural number. This space is equipped with a natural product topology defined in Topology˙ZF˙3.

Let's recall first that the sum of elements of a group is an element of the group.

lemma (in topgroup) sum˙list˙in˙group:
  assumes n ∈ nat and x: succ(n)→G
  shows (∑x) ∈ G
⟨*proof*⟩

In this context x+y is the same as the value of the group operation on the elements x and y. Normally we shouldn't need to state this a s separate lemma.

lemma (in topgroup) grop˙def1: shows f⟨x,y⟩ = x+y ⟨*proof*⟩

Another theorem from Semigroup˙ZF theory that is useful to have in the additive notation.

lemma (in topgroup) shorter˙set˙add:
  assumes n ∈ nat and x: succ(succ(n))→G
  shows (∑x) = (∑Init(x)) + (x(succ(n)))
⟨*proof*⟩

Sum is a continuous function in the product topology.

theorem (in topgroup) sum˙continuous: assumes n ∈ nat
  shows IsContinuous(SeqProductTopology(succ(n),T),T,–⟨x,∑x⟩.x∈succ(n)→G″)
  ⟨*proof*⟩
end

73 Topological groups 1

theory TopologicalGroup˙ZF˙1 imports TopologicalGroup˙ZF Topology˙ZF˙properties˙2

begin

This theory deals with some topological properties of topological groups.

73.1 Separation properties of topological groups

The topological groups have very specific properties. For instance, G is T_0 iff it is T_3.

theorem(in topgroup) cl˙point:
  assumes x∈G
  shows cl(–x″) = ($\bigcap$H∈$\mathcal{N}_0$. x+H)
⟨*proof*⟩

We prove the equivalence between T_0 and T_1 first.

theorem (in topgroup) neu˙closed˙imp˙T1:
  assumes –**0**″–is closed in″T
  shows T–is T_1″
⟨*proof*⟩

theorem (in topgroup) T0˙imp˙neu˙closed:
  assumes T–is T_0″
  shows –**0**″–is closed in″T
⟨*proof*⟩

73.2 Existence of nice neighbourhoods.

lemma (in topgroup) exist˙basehoods˙closed:
  assumes U∈$\mathcal{N}_0$
  shows ∃V∈$\mathcal{N}_0$. cl(V)⊆U
⟨*proof*⟩

73.3 Rest of separation axioms

theorem(in topgroup) T1˙imp˙T2:
  assumes T–is T_1″
  shows T–is T_2″
⟨*proof*⟩

Here follow some auxiliary lemmas.

lemma (in topgroup) trans˙closure:
  assumes x∈G A⊆G
  shows cl(x+A)=x+cl(A)
⟨*proof*⟩

lemma (in topgroup) trans˙interior2: assumes A1: g∈G and A2: A⊆G
  shows int(A)+g = int(A+g)
⟨*proof*⟩

lemma (in topgroup) trans˙closure2:
  assumes x∈G A⊆G
  shows cl(A+x)=cl(A)+x
⟨*proof*⟩

lemma (in topgroup) trans˙subset:
  assumes A⊆((-x)+B)x∈GA⊆GB⊆G
  shows x+A⊆B
⟨*proof*⟩

Every topological group is regular, and hence T_3. The proof is in the next section, since it uses local properties.

73.4 Local properties

In a topological group, all local properties depend only on the neighbourhoods of the neutral element; when considering topological properties. The next result of regularity, will use this idea, since translations preserve closed sets.

lemma (in topgroup) local˙iff˙neutral:
  assumes ∀U∈T∩$\mathcal{N}_0$. ∃N∈$\mathcal{N}_0$. N⊆U∧ P(N,T) ∀N∈Pow(G). ∀x∈G. P(N,T) ⟶ P(x+N,T)
  shows T–is locally″P
⟨*proof*⟩

lemma (in topgroup) trans˙closed:
  assumes A–is closed in″Tx∈G
  shows (x+A)–is closed in″T
⟨*proof*⟩

As it is written in the previous section, every topological group is regular.

theorem (in topgroup) topgroup˙reg:
  shows T–is regular″
⟨*proof*⟩

The promised corollary follows:

corollary (in topgroup) T2˙imp˙T3:
  assumes T–is T_2″
  shows T–is T_3″ ⟨*proof*⟩

end

74 Topological groups - uniformity

theory TopologicalGroup˙Uniformity˙ZF imports TopologicalGroup˙ZF UniformSpace˙ZF˙1

begin

Each topological group is a uniform space. This theory is about the unifomities that are naturally defined by a topological group structure.

74.1 Natural uniformities in topological groups: definitions and notation

There are two basic uniformities that can be defined on a topological group.

Definition of left uniformity

definition (in topgroup) leftUniformity
where leftUniformity ≡ –V∈Pow(G×G).∃ U∈ $\mathcal{N}_0$. –⟨s,t⟩∈G×G. (-s)+t ∈U″ ⊆ V″

Definition of right uniformity

definition (in topgroup) rightUniformity
where rightUniformity ≡ –V∈Pow(G×G).∃ U∈ $\mathcal{N}_0$. –⟨s,t⟩∈G×G. s+(-t) ∈U″⊆ V″

Right and left uniformities are indeed uniformities.

lemma (in topgroup) side˙uniformities:
shows leftUniformity –is a uniformity on″ G and rightUniformity –is a uniformity on″ G
⟨*proof*⟩

The topologies generated by the right and left uniformities are the original group topology.

lemma (in topgroup) top˙generated˙side˙uniformities:
shows UniformTopology(leftUniformity,G) = T and UniformTopology(rightUniformity,G) = T
⟨*proof*⟩

The side uniformities are called this way because of how they affect left and right translations. In the next lemma we show that left translations are uniformly continuous with respect to the left uniformity.

lemma (in topgroup) left˙mult˙uniformity: assumes x∈G
shows
LeftTranslation(G,f,x) –is uniformly continuous between″ leftUniformity –and″ leftUniformity
⟨*proof*⟩

Right translations are uniformly continuous with respect to the right uniformity.

lemma (in topgroup) right˙mult˙uniformity: assumes x∈G
shows
RightTranslation(G,f,x) –is uniformly continuous between″ rightUniformity –and″ rightUniformity
⟨*proof*⟩

The third uniformity important on topological groups is called the uniformity of Roelcke.

definition(in topgroup) roelckeUniformity
where roelckeUniformity ≡ –V∈Pow(G×G). ∃ U∈ $\mathcal{N}_0$. –⟨s,t⟩∈G×G. t ∈(U+s)+U″⊆ V″

The Roelcke uniformity is indeed a uniformity on the group.

lemma (in topgroup) roelcke˙uniformity:
shows roelckeUniformity –is a uniformity on″ G
⟨*proof*⟩

The topology given by the roelcke uniformity is the original topology

lemma (in topgroup) top˙generated˙roelcke˙uniformity:
shows UniformTopology(roelckeUniformity,G) = T
⟨*proof*⟩

The inverse map is uniformly continuous in the Roelcke uniformity

theorem (in topgroup) inv˙uniform˙roelcke:
shows
GroupInv(G,f) –is uniformly continuous between″ roelckeUniformity –and″ roelckeUniformity
⟨*proof*⟩

end

75 Topological groups 2

theory TopologicalGroup˙ZF˙2 imports Topology˙ZF˙8 TopologicalGroup˙ZF Group˙ZF˙2
begin

This theory deals with quotient topological groups.

75.1 Quotients of topological groups

The quotient topology given by the quotient group equivalent relation, has an open quotient map.

theorem(in topgroup) quotient˙map˙topgroup˙open:
assumes IsAsubgroup(H,f) A∈T
defines r ≡ QuotientGroupRel(G,f,H)
shows –⟨b,r–b″⟩. b∈⋃T″A∈(T–quotient by″r)
⟨*proof*⟩

A quotient of a topological group is just a quotient group with an appropiate topology that makes product and inverse continuous.

theorem (in topgroup) quotient˙top˙group˙F˙cont:
assumes IsAnormalSubgroup(G,f,H)

defines r ≡ QuotientGroupRel(G,f,H)
defines F ≡ QuotientGroupOp(G,f,H)
shows IsContinuous(ProductTopology(T–quotient by''r,T–quotient by''r),T–quotient by''r,F)
⟨*proof*⟩

lemma (in group0) Group_ZF_2_4_L8:
assumes IsAnormalSubgroup(G,P,H)
defines r ≡ QuotientGroupRel(G,P,H)
and F ≡ QuotientGroupOp(G,P,H)
shows GroupInv(G//r,F):G//r→G//r
⟨*proof*⟩

theorem (in topgroup) quotient_top_group_INV_cont:
assumes IsAnormalSubgroup(G,f,H)
defines r ≡ QuotientGroupRel(G,f,H)
defines F ≡ QuotientGroupOp(G,f,H)
shows IsContinuous(T–quotient by''r,T–quotient by''r,GroupInv(G//r,F))
⟨*proof*⟩

Finally we can prove that quotient groups of topological groups are topological groups.

theorem(in topgroup) quotient_top_group:
assumes IsAnormalSubgroup(G,f,H)
defines r ≡ QuotientGroupRel(G,f,H)
defines F ≡ QuotientGroupOp(G,f,H)
shows IsAtopologicalGroup(–quotient by''r,F)
⟨*proof*⟩

end

76 Topological groups 3

theory TopologicalGroup_ZF_3 imports Topology_ZF_10 TopologicalGroup_ZF_2 TopologicalGroup_ZF_1
Group_ZF_4

begin

This theory deals with topological properties of subgroups, quotient groups and relations between group theorical properties and topological properties.

76.1 Subgroups topologies

The closure of a subgroup is a subgroup.

theorem (in topgroup) closure_subgroup:

assumes IsAsubgroup(H,f)
shows IsAsubgroup(cl(H),f)
⟨*proof*⟩

The closure of a normal subgroup is normal.

theorem (in topgroup) normal˙subg:
assumes IsAnormalSubgroup(G,f,H)
shows IsAnormalSubgroup(G,f,cl(H))
⟨*proof*⟩

Every open subgroup is also closed.

theorem (in topgroup) open˙subgroup˙closed:
assumes IsAsubgroup(H,f) H∈T
shows H–is closed in″T
⟨*proof*⟩

Any subgroup with non-empty interior is open.

theorem (in topgroup) clopen˙or˙emptyInt:
assumes IsAsubgroup(H,f) int(H)≠0
shows H∈T
⟨*proof*⟩

In conclusion, a subgroup is either open or has empty interior.

corollary(in topgroup) emptyInterior˙xor˙op:
assumes IsAsubgroup(H,f)
shows (int(H)=0) Xor (H∈T)
⟨*proof*⟩

Then no connected topological groups has proper subgroups with non-empty interior.

corollary(in topgroup) connected˙emptyInterior:
assumes IsAsubgroup(H,f) T–is connected″
shows (int(H)=0) Xor (H=G)
⟨*proof*⟩

Every locally-compact subgroup of a T_0 group is closed.

theorem (in topgroup) loc˙compact˙T0˙closed:
assumes IsAsubgroup(H,f) (T–restricted to″H)–is locally-compact″ T–is T_0″
shows H–is closed in″T
⟨*proof*⟩

We can always consider a factor group which is T_2.

theorem(in topgroup) factor˙haus:
shows (T–quotient by″QuotientGroupRel(G,f,cl(–**0**″)))–is T_2″
⟨*proof*⟩

end

77 Metamath introduction

theory MMI˙prelude imports Order˙ZF˙1

begin

Metamath's set.mm features a large (over 8000) collection of theorems proven in the ZFC set theory. This theory is part of an attempt to translate those theorems to Isar so that they are available for Isabelle/ZF users. A total of about 1200 assertions have been translated, 600 of that with proofs (the rest was proven automatically by Isabelle). The translation was done with the support of the mmisar tool, whose source is included in the IsarMathLib distributions prior to version 1.6.4. The translation tool was doing about 99 percent of work involved, with the rest mostly related to the difference between Isabelle/ZF and Metamath metalogics. Metamath uses Tarski-Megill metalogic that does not have a notion of bound variables (see http://planetx.cc.vt.edu/AsteroidMeta/Distinctors˙vs˙binders for details and discussion). The translation project is closed now as I decided that it was too boring and tedious even with the support of mmisar software. Also, the translated proofs are not as readable as native Isar proofs which goes against IsarMathLib philosophy.

77.1 Importing from Metamath - how is it done

We are interested in importing the theorems about complex numbers that start from the "recnt" theorem on. This is done mostly automatically by the mmisar tool that is included in the IsarMathLib distributions prior to version 1.6.4. The tool works as follows:

First it reads the list of (Metamath) names of theorems that are already imported to IsarMathlib ("known theorems") and the list of theorems that are intended to be imported in this session ("new theorems"). The new theorems are consecutive theorems about complex numbers as they appear in the Metamath database. Then mmisar creates a "Metamath script" that contains Metamath commands that open a log file and put the statements and proofs of the new theorems in that file in a readable format. The tool writes this script to a disk file and executes metamath with standard input redirected from that file. Then the log file is read and its contents converted to the Isar format. In Metamath, the proofs of theorems about complex numbers depend only on 28 axioms of complex numbers and some basic logic and set theory theorems. The tool finds which of these dependencies are not known yet and repeats the process of getting their statements from Metamath as with the new theorems. As a result of this process mmisar creates files new_theorems.thy, new_deps.thy and new_known_theorems.txt. The file new_theorems.thy contains the theorems (with proofs) imported from Metamath in this session. These theorems are added (by hand) to the current MMI˙Complex˙ZF˙x.thy file. The file new_deps.thy contains the statements of new dependencies with generic proofs "by auto". These are added to the MMI˙logic˙and˙sets.thy. Most of the dependencies can be proven automatically by Isabelle. However, some manual work has to be done for the dependencies that Isabelle can not prove by itself and to correct problems related to the fact that Metamath uses a metalogic based on distinct variable constraints (Tarski-Megill metalogic), rather than an explicit notion of free and bound variables.

The old list of known theorems is replaced by the new list and mmisar is ready to convert the next batch of new theorems. Of course this rarely works in practice without tweaking the mmisar source files every time a new batch is processed.

77.2 The context for Metamath theorems

We list the Metamth's axioms of complex numbers and define notation here.

The next definition is what Metamath $X \in V$ is translated to. I am not sure why it works, probably because Isabelle does a type inference and the "=" sign indicates that both sides are sets.

definition
  IsASet :: i⇒o (˙ isASet [90] 90) where

  IsASet˙def[simp]: X isASet ≡ X = X

The next locale sets up the context to which Metamath theorems about complex numbers are imported. It assumes the axioms of complex numbers and defines the notation used for complex numbers.

One of the problems with importing theorems from Metamath is that Metamath allows direct infix notation for binary operations so that the notation afb is allowed where f is a function (that is, a set of pairs). To my knowledge, Isar allows only notation f⟨a,b⟩ with a possibility of defining a syntax say a + b to mean the same as f⟨a,b⟩ (please correct me if I am wrong here). This is why we have two objects for addition: one called caddset that represents the binary function, and the second one called ca which defines the a + b notation for caddset⟨a,b⟩. The same applies to multiplication of real numbers.

Another difficulty is that Metamath allows to define sets with syntax $\{x|p\}$ where p is some formula that (usually) depends on x. Isabelle allows the set comprehension like this only as a subset of another set i.e. $\{x \in A.p(x)\}$. This forces us to have a sligtly different definition of (complex) natural numbers, requiring explicitly that natural numbers is a subset of reals. Because of that, the proofs of Metamath theorems that reference the definition directly can not be imported.

```
locale MMIsar0 =
  fixes real (ℝ)
  fixes complex (ℂ)
  fixes one (1)
  fixes zero (0)
  fixes iunit (i)
  fixes caddset (+)
  fixes cmulset (·)
  fixes lessrrel (<ℝ)

  fixes ca (infixl + 69)
  defines ca˙def: a + b ≡ +⟨a,b⟩
  fixes cm (infixl · 71)
  defines cm˙def: a · b ≡ ·⟨a,b⟩
  fixes sub (infixl - 69)
  defines sub˙def: a - b ≡ ⋃ – x ∈ ℂ. b + x = a ″
  fixes cneg (-˙ 95)
  defines cneg˙def: - a ≡ 0 - a
  fixes cdiv (infixl / 70)
  defines cdiv˙def: a / b ≡ ⋃ – x ∈ ℂ. b · x = a ″
  fixes cpnf (+∞)
  defines cpnf˙def: +∞ ≡ ℂ
  fixes cmnf (–∞)
  defines cmnf˙def: –∞ ≡ –ℂ″
  fixes cxr (ℝ*)
  defines cxr˙def: ℝ* ≡ ℝ ∪ –+∞,–∞″
  fixes cxn (ℕ)
  defines cxn˙def: ℕ ≡ ⋂ –N ∈ Pow(ℝ). 1 ∈ N ∧ (∀n. n∈N ⟶ n+1 ∈ N)″
  fixes lessr (infix <ℝ 68)
  defines lessr˙def: a <ℝ b ≡ ⟨a,b⟩ ∈ <ℝ
  fixes cltrrset (¡)
  defines cltrrset˙def:
  ¡ ≡ (<ℝ ∩ ℝ×ℝ) ∪ –⟨–∞,+∞⟩″ ∪
  (ℝ×–+∞″) ∪ (––∞″×ℝ )
  fixes cltrr (infix ¡ 68)
  defines cltrr˙def: a ¡ b ≡ ⟨a,b⟩ ∈ ¡
  fixes convcltrr (infix ¿ 68)
  defines convcltrr˙def: a ¿ b ≡ ⟨a,b⟩ ∈ converse(¡)
  fixes lsq (infix ≤ 68)
  defines lsq˙def: a ≤ b ≡ ¬ (b ¡ a)
  fixes two (2)
  defines two˙def: 2 ≡ 1+1
  fixes three (3)
  defines three˙def: 3 ≡ 2+1
  fixes four (4)
  defines four˙def: 4 ≡ 3+1
  fixes five (5)
  defines five˙def: 5 ≡ 4+1
  fixes six (6)
```

defines six˙def: $\mathbf{6} \equiv \mathbf{5}+\mathbf{1}$
fixes seven ($\mathbf{7}$)
defines seven˙def: $\mathbf{7} \equiv \mathbf{6}+\mathbf{1}$
fixes eight ($\mathbf{8}$)
defines eight˙def: $\mathbf{8} \equiv \mathbf{7}+\mathbf{1}$
fixes nine ($\mathbf{9}$)
defines nine˙def: $\mathbf{9} \equiv \mathbf{8}+\mathbf{1}$

assumes MMI˙pre˙axlttri:
$A \in \mathbb{R} \wedge B \in \mathbb{R} \longrightarrow (A <_{\mathbb{R}} B \longleftrightarrow \neg(A=B \vee B <_{\mathbb{R}} A))$
assumes MMI˙pre˙axlttrn:
$A \in \mathbb{R} \wedge B \in \mathbb{R} \wedge C \in \mathbb{R} \longrightarrow ((A <_{\mathbb{R}} B \wedge B <_{\mathbb{R}} C) \longrightarrow A <_{\mathbb{R}} C)$
assumes MMI˙pre˙axltadd:
$A \in \mathbb{R} \wedge B \in \mathbb{R} \wedge C \in \mathbb{R} \longrightarrow (A <_{\mathbb{R}} B \longrightarrow C+A <_{\mathbb{R}} C+B)$
assumes MMI˙pre˙axmulgt0:
$A \in \mathbb{R} \wedge B \in \mathbb{R} \longrightarrow (\mathbf{0} <_{\mathbb{R}} A \wedge \mathbf{0} <_{\mathbb{R}} B \longrightarrow \mathbf{0} <_{\mathbb{R}} A \cdot B)$
assumes MMI˙pre˙axsup:
$A \subseteq \mathbb{R} \wedge A \neq 0 \wedge (\exists x \in \mathbb{R}. \forall y \in A. y <_{\mathbb{R}} x) \longrightarrow$
$(\exists x \in \mathbb{R}. (\forall y \in A. \neg(x <_{\mathbb{R}} y)) \wedge (\forall y \in \mathbb{R}. (y <_{\mathbb{R}} x \longrightarrow (\exists z \in A. y <_{\mathbb{R}} z))))$
assumes MMI˙axresscn: $\mathbb{R} \subseteq \mathbb{C}$
assumes MMI˙ax1ne0: $\mathbf{1} \neq \mathbf{0}$
assumes MMI˙axcnex: $\mathbb{C}$ isASet
assumes MMI˙axaddopr: $+ : (\mathbb{C} \times \mathbb{C}) \to \mathbb{C}$
assumes MMI˙axmulopr: $\cdot : (\mathbb{C} \times \mathbb{C}) \to \mathbb{C}$
assumes MMI˙axmulcom: $A \in \mathbb{C} \wedge B \in \mathbb{C} \longrightarrow A \cdot B = B \cdot A$
assumes MMI˙axaddcl: $A \in \mathbb{C} \wedge B \in \mathbb{C} \longrightarrow A + B \in \mathbb{C}$
assumes MMI˙axmulcl: $A \in \mathbb{C} \wedge B \in \mathbb{C} \longrightarrow A \cdot B \in \mathbb{C}$
assumes MMI˙axdistr:
$A \in \mathbb{C} \wedge B \in \mathbb{C} \wedge C \in \mathbb{C} \longrightarrow A\cdot(B + C) = A\cdot B + A\cdot C$
assumes MMI˙axaddcom: $A \in \mathbb{C} \wedge B \in \mathbb{C} \longrightarrow A + B = B + A$
assumes MMI˙axaddass:
$A \in \mathbb{C} \wedge B \in \mathbb{C} \wedge C \in \mathbb{C} \longrightarrow A + B + C = A + (B + C)$
assumes MMI˙axmulass:
$A \in \mathbb{C} \wedge B \in \mathbb{C} \wedge C \in \mathbb{C} \longrightarrow A \cdot B \cdot C = A \cdot (B \cdot C)$
assumes MMI˙ax1re: $\mathbf{1} \in \mathbb{R}$
assumes MMI˙axi2m1: $i \cdot i + \mathbf{1} = \mathbf{0}$
assumes MMI˙ax0id: $A \in \mathbb{C} \longrightarrow A + \mathbf{0} = A$
assumes MMI˙axicn: $i \in \mathbb{C}$
assumes MMI˙axnegex: $A \in \mathbb{C} \longrightarrow (\exists\ x \in \mathbb{C}. (A + x) = \mathbf{0})$
assumes MMI˙axrecex: $A \in \mathbb{C} \wedge A \neq \mathbf{0} \longrightarrow (\exists\ x \in \mathbb{C}. A \cdot x = \mathbf{1})$
assumes MMI˙ax1id: $A \in \mathbb{C} \longrightarrow A \cdot \mathbf{1} = A$
assumes MMI˙axaddrcl: $A \in \mathbb{R} \wedge B \in \mathbb{R} \longrightarrow A + B \in \mathbb{R}$
assumes MMI˙axmulrcl: $A \in \mathbb{R} \wedge B \in \mathbb{R} \longrightarrow A \cdot B \in \mathbb{R}$
assumes MMI˙axrnegex: $A \in \mathbb{R} \longrightarrow (\exists\ x \in \mathbb{R}. A + x = \mathbf{0})$
assumes MMI˙axrrecex: $A \in \mathbb{R} \wedge A \neq \mathbf{0} \longrightarrow (\exists\ x \in \mathbb{R}. A \cdot x = \mathbf{1})$

end

78 Logic and sets in Metamatah

theory MMI˙logic˙and˙sets imports MMI˙prelude

begin

78.1 Basic Metamath theorems

This section contains Metamath theorems that the more advanced theorems from MMIsar.thy depend on. Most of these theorems are proven automatically by Isabelle, some have to be proven by hand and some have to be modified to convert from Tarski-Megill metalogic used by Metamath to one based on explicit notion of free and bound variables.

lemma MMI˙ax˙mp: assumes φ and $\varphi \longrightarrow \psi$ shows ψ
⟨*proof*⟩

lemma MMI˙sseli: assumes A1: A $\subseteq$ B
shows C $\in$ A $\longrightarrow$ C $\in$ B
⟨*proof*⟩

lemma MMI˙sselii: assumes A1: A $\subseteq$ B and
A2: C $\in$ A
shows C $\in$ B
⟨*proof*⟩

lemma MMI˙syl: assumes A1: $\varphi \longrightarrow$ ps and
A2: ps $\longrightarrow$ ch
shows $\varphi \longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙elimhyp: assumes A1: A = if (φ , A , B) $\longrightarrow$ ($\varphi \longleftrightarrow \psi$) and
A2: B = if (φ , A , B) $\longrightarrow$ (ch $\longleftrightarrow \psi$) and
A3: ch
shows ψ
⟨*proof*⟩

lemma MMI˙neeq1:
shows A = B $\longrightarrow$ (A $\neq$ C $\longleftrightarrow$ B $\neq$ C)
⟨*proof*⟩

lemma MMI˙mp2: assumes A1: φ and
A2: ψ and
A3: $\varphi \longrightarrow$ ($\psi \longrightarrow$ chi)
shows chi
⟨*proof*⟩

lemma MMI˙xpex: assumes A1: A isASet and
A2: B isASet

shows (A $\times$ B) isASet
$\langle proof \rangle$

lemma MMI˙fex:
shows
A $\in$ C $\longrightarrow$ (F : A $\rightarrow$ B $\longrightarrow$ F isASet)
A isASet $\longrightarrow$ (F : A $\rightarrow$ B $\longrightarrow$ F isASet)
$\langle proof \rangle$

lemma MMI˙3eqtr4d: assumes A1: φ $\longrightarrow$ A = B and
A2: φ $\longrightarrow$ C = A and
A3: φ $\longrightarrow$ D = B
shows φ $\longrightarrow$ C = D
$\langle proof \rangle$

lemma MMI˙3coml: assumes A1: (φ $\wedge$ ψ $\wedge$ chi) $\longrightarrow$ th
shows (ψ $\wedge$ chi $\wedge$ φ) $\longrightarrow$ th
$\langle proof \rangle$

lemma MMI˙sylan: assumes A1: (φ $\wedge$ ψ) $\longrightarrow$ chi and
A2: th $\longrightarrow$ φ
shows (th $\wedge$ ψ) $\longrightarrow$ chi
$\langle proof \rangle$

lemma MMI˙3impa: assumes A1: ((φ $\wedge$ ψ) $\wedge$ chi) $\longrightarrow$ th
shows (φ $\wedge$ ψ $\wedge$ chi) $\longrightarrow$ th
$\langle proof \rangle$

lemma MMI˙3adant2: assumes A1: (φ $\wedge$ ψ) $\longrightarrow$ chi
shows (φ $\wedge$ th $\wedge$ ψ) $\longrightarrow$ chi
$\langle proof \rangle$

lemma MMI˙3adant1: assumes A1: (φ $\wedge$ ψ) $\longrightarrow$ chi
shows (th $\wedge$ φ $\wedge$ ψ) $\longrightarrow$ chi
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙opreq12d: assumes A1: φ $\longrightarrow$ A = B and
A2: φ $\longrightarrow$ C = D
shows
φ $\longrightarrow$ (A + C) = (B + D)
φ $\longrightarrow$ (A $\cdot$ C) = (B $\cdot$ D)
φ $\longrightarrow$ (A - C) = (B - D)
φ $\longrightarrow$ (A / C) = (B / D)
$\langle proof \rangle$

lemma MMI˙mp2an: assumes A1: φ and
A2: ψ and
A3: (φ $\wedge$ ψ) $\longrightarrow$ chi
shows chi

⟨*proof*⟩

lemma MMI˙mp3an: assumes A1: φ and
A2: ψ and
A3: ch and
A4: ($\varphi \wedge \psi \wedge$ ch) $\longrightarrow \vartheta$
shows ϑ
⟨*proof*⟩

lemma MMI˙eqeltrr: assumes A1: A = B and
A2: A $\in$ C
shows B $\in$ C
⟨*proof*⟩

lemma MMI˙eqtr: assumes A1: A = B and
A2: B = C
shows A = C
⟨*proof*⟩

lemma MMI˙impbi: assumes A1: $\varphi \longrightarrow \psi$ and
A2: $\psi \longrightarrow \varphi$
shows $\varphi \longleftrightarrow \psi$
⟨*proof*⟩

lemma MMI˙mp3an3: assumes A1: ch and
A2: ($\varphi \wedge \psi \wedge$ ch) $\longrightarrow \vartheta$
shows ($\varphi \wedge \psi$) $\longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙eqeq12d: assumes A1: $\varphi \longrightarrow$ A = B and
A2: $\varphi \longrightarrow$ C = D
shows $\varphi \longrightarrow$ (A = C $\longleftrightarrow$ B = D)
⟨*proof*⟩

lemma MMI˙mpan2: assumes A1: ψ and
A2: ($\varphi \wedge \psi$) $\longrightarrow$ ch
shows $\varphi \longrightarrow$ ch
⟨*proof*⟩

lemma (in MMIsar0) MMI˙opreq2:
shows
A = B $\longrightarrow$ (C + A) = (C + B)
A = B $\longrightarrow$ (C $\cdot$ A) = (C $\cdot$ B)
A = B $\longrightarrow$ (C - A) = (C - B)
A = B $\longrightarrow$ (C / A) = (C / B)
⟨*proof*⟩

lemma MMI˙syl5bir: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow$ ch) and
A2: $\vartheta \longrightarrow$ ch
shows $\varphi \longrightarrow (\vartheta \longrightarrow \psi)$
⟨*proof*⟩

lemma MMI˙adantr: assumes A1: $\varphi \longrightarrow \psi$
shows ($\varphi \wedge$ ch) $\longrightarrow \psi$
⟨*proof*⟩

lemma MMI˙mpan: assumes A1: φ and
A2: ($\varphi \wedge \psi$) $\longrightarrow$ ch
shows $\psi \longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙eqeq1d: assumes A1: $\varphi \longrightarrow$ A = B
shows $\varphi \longrightarrow$ (A = C $\longleftrightarrow$ B = C)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙opreq1:
shows
A = B $\longrightarrow$ (A $\cdot$ C) = (B $\cdot$ C)
A = B $\longrightarrow$ (A + C) = (B + C)
A = B $\longrightarrow$ (A - C) = (B - C)
A = B $\longrightarrow$ (A / C) = (B / C)
⟨*proof*⟩

lemma MMI˙syl6eq: assumes A1: $\varphi \longrightarrow$ A = B and
A2: B = C
shows $\varphi \longrightarrow$ A = C
⟨*proof*⟩

lemma MMI˙syl6bi: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow$ ch) and
A2: ch $\longrightarrow \vartheta$
shows $\varphi \longrightarrow (\psi \longrightarrow \vartheta)$
⟨*proof*⟩

lemma MMI˙imp: assumes A1: $\varphi \longrightarrow (\psi \longrightarrow$ ch)
shows ($\varphi \wedge \psi$) $\longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙sylibd: assumes A1: $\varphi \longrightarrow (\psi \longrightarrow$ ch) and
A2: $\varphi \longrightarrow$ (ch $\longleftrightarrow \vartheta$)
shows $\varphi \longrightarrow (\psi \longrightarrow \vartheta)$
⟨*proof*⟩

lemma MMI˙ex: assumes A1: ($\varphi \wedge \psi$) $\longrightarrow$ ch
shows $\varphi \longrightarrow (\psi \longrightarrow$ ch)
⟨*proof*⟩

lemma MMI˙r19˙23aiv: assumes A1: $\forall$x. (x $\in$ A $\longrightarrow$ (φ(x) $\longrightarrow$ ψ))
shows ($\exists$ x $\in$ A . φ(x)) $\longrightarrow$ ψ
⟨*proof*⟩

lemma MMI˙bitr: assumes A1: $\varphi \longleftrightarrow \psi$ and
A2: $\psi \longleftrightarrow$ ch
shows $\varphi \longleftrightarrow$ ch
⟨*proof*⟩

lemma MMI˙eqeq12i: assumes A1: A = B and
A2: C = D
shows A = C $\longleftrightarrow$ B = D
⟨*proof*⟩

lemma MMI˙dedth3h:
assumes A1: A = if (φ , A , D) $\longrightarrow$ ($\vartheta \longleftrightarrow$ ta) and
A2: B = if (ψ , B , R) $\longrightarrow$ (ta $\longleftrightarrow$ et) and
A3: C = if (ch , C , S) $\longrightarrow$ (et $\longleftrightarrow$ ze) and
A4: ze
shows ($\varphi \wedge \psi \wedge$ ch) $\longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙bibi1d: assumes A1: $\varphi \longrightarrow$ ($\psi \longleftrightarrow$ ch)
shows $\varphi \longrightarrow$ (($\psi \longleftrightarrow \vartheta$) $\longleftrightarrow$ (ch $\longleftrightarrow \vartheta$))
⟨*proof*⟩

lemma MMI˙eqeq1:
shows A = B $\longrightarrow$ (A = C $\longleftrightarrow$ B = C)
⟨*proof*⟩

lemma MMI˙bibi12d: assumes A1: $\varphi \longrightarrow$ ($\psi \longleftrightarrow$ ch) and
A2: $\varphi \longrightarrow$ ($\vartheta \longleftrightarrow$ ta)
shows $\varphi \longrightarrow$ (($\psi \longleftrightarrow \vartheta$) $\longleftrightarrow$ (ch $\longleftrightarrow$ ta))
⟨*proof*⟩

lemma MMI˙eqeq2d: assumes A1: $\varphi \longrightarrow$ A = B
shows $\varphi \longrightarrow$ (C = A $\longleftrightarrow$ C = B)
⟨*proof*⟩

lemma MMI˙eqeq2:
shows A = B $\longrightarrow$ (C = A $\longleftrightarrow$ C = B)
⟨*proof*⟩

lemma MMI˙elimel: assumes A1: B $\in$ C
shows if (A $\in$ C , A , B) $\in$ C
⟨*proof*⟩

lemma MMI˙3adant3: assumes A1: ($\varphi \wedge \psi$) $\longrightarrow$ ch
shows ($\varphi \wedge \psi \wedge \vartheta$) $\longrightarrow$ ch

$\langle proof \rangle$

lemma MMI˙bitr3d: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow ch)$ and
A2: $\varphi \longrightarrow (\psi \longleftrightarrow \vartheta)$
shows $\varphi \longrightarrow (ch \longleftrightarrow \vartheta)$
$\langle proof \rangle$

lemma MMI˙3eqtr3d: assumes A1: $\varphi \longrightarrow A = B$ and
A2: $\varphi \longrightarrow A = C$ and
A3: $\varphi \longrightarrow B = D$
shows $\varphi \longrightarrow C = D$
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙opreq1d: assumes A1: $\varphi \longrightarrow A = B$
shows
$\varphi \longrightarrow (A + C) = (B + C)$
$\varphi \longrightarrow (A - C) = (B - C)$
$\varphi \longrightarrow (A \cdot C) = (B \cdot C)$
$\varphi \longrightarrow (A / C) = (B / C)$
$\langle proof \rangle$

lemma MMI˙3com12: assumes A1: $(\varphi \wedge \psi \wedge ch) \longrightarrow \vartheta$
shows $(\psi \wedge \varphi \wedge ch) \longrightarrow \vartheta$
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙opreq2d: assumes A1: $\varphi \longrightarrow A = B$
shows
$\varphi \longrightarrow (C + A) = (C + B)$
$\varphi \longrightarrow (C - A) = (C - B)$
$\varphi \longrightarrow (C \cdot A) = (C \cdot B)$
$\varphi \longrightarrow (C / A) = (C / B)$
$\langle proof \rangle$

lemma MMI˙3com23: assumes A1: $(\varphi \wedge \psi \wedge ch) \longrightarrow \vartheta$
shows $(\varphi \wedge ch \wedge \psi) \longrightarrow \vartheta$
$\langle proof \rangle$

lemma MMI˙3expa: assumes A1: $(\varphi \wedge \psi \wedge ch) \longrightarrow \vartheta$
shows $((\varphi \wedge \psi) \wedge ch) \longrightarrow \vartheta$
$\langle proof \rangle$

lemma MMI˙adantrr: assumes A1: $(\varphi \wedge \psi) \longrightarrow ch$
shows $(\varphi \wedge (\psi \wedge \vartheta)) \longrightarrow ch$
$\langle proof \rangle$

lemma MMI˙3expb: assumes A1: $(\varphi \wedge \psi \wedge ch) \longrightarrow \vartheta$
shows $(\varphi \wedge (\psi \wedge ch)) \longrightarrow \vartheta$

⟨*proof*⟩

lemma MMI˙an4s: assumes A1: $((\varphi \wedge \psi) \wedge (\text{ch} \wedge \vartheta)) \longrightarrow \tau$
shows $((\varphi \wedge \text{ch}) \wedge (\psi \wedge \vartheta)) \longrightarrow \tau$
⟨*proof*⟩

lemma MMI˙eqtrd: assumes A1: $\varphi \longrightarrow A = B$ and
A2: $\varphi \longrightarrow B = C$
shows $\varphi \longrightarrow A = C$
⟨*proof*⟩

lemma MMI˙ad2ant2l: assumes A1: $(\varphi \wedge \psi) \longrightarrow \text{ch}$
shows $((\vartheta \wedge \varphi) \wedge (\tau \wedge \psi)) \longrightarrow \text{ch}$
⟨*proof*⟩

lemma MMI˙pm3˙2i: assumes A1: φ and
A2: ψ
shows $\varphi \wedge \psi$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙opreq2i: assumes A1: A = B
shows
$(C + A) = (C + B)$
$(C - A) = (C - B)$
$(C \cdot A) = (C \cdot B)$
⟨*proof*⟩

lemma MMI˙mpbir2an: assumes A1: $\varphi \longleftrightarrow (\psi \wedge \text{ch})$ and
A2: ψ and
A3: ch
shows φ
⟨*proof*⟩

lemma MMI˙reu4: assumes A1: $\forall x\ y.\ x = y \longrightarrow (\varphi(x) \longleftrightarrow \psi(y))$
shows $(\exists! \ x .\ x \in A \wedge \varphi(x)) \longleftrightarrow$
$((\exists\ x \in A .\ \varphi(x)) \wedge (\forall\ x \in A .\ \forall\ y \in A .$
$((\varphi(x) \wedge \psi(y)) \longrightarrow x = y)))$
⟨*proof*⟩

lemma MMI˙risset:
shows $A \in B \longleftrightarrow (\exists\ x \in B .\ x = A)$
⟨*proof*⟩

lemma MMI˙sylib: assumes A1: $\varphi \longrightarrow \psi$ and
A2: $\psi \longleftrightarrow \text{ch}$
shows $\varphi \longrightarrow \text{ch}$
⟨*proof*⟩

lemma MMI˙mp3an13: assumes A1: φ and
A2: ch and
A3: ($\varphi \wedge \psi \wedge$ ch) $\longrightarrow \vartheta$
shows $\psi \longrightarrow \vartheta$
$\langle proof \rangle$

lemma MMI˙eqcomd: assumes A1: $\varphi \longrightarrow$ A = B
shows $\varphi \longrightarrow$ B = A
$\langle proof \rangle$

lemma MMI˙sylan9eqr: assumes A1: $\varphi \longrightarrow$ A = B and
A2: $\psi \longrightarrow$ B = C
shows ($\psi \wedge \varphi$) $\longrightarrow$ A = C
$\langle proof \rangle$

lemma MMI˙exp32: assumes A1: ($\varphi \wedge$ ($\psi \wedge$ ch)) $\longrightarrow \vartheta$
shows $\varphi \longrightarrow$ ($\psi \longrightarrow$ (ch $\longrightarrow \vartheta$))
$\langle proof \rangle$

lemma MMI˙impcom: assumes A1: $\varphi \longrightarrow$ ($\psi \longrightarrow$ ch)
shows ($\psi \wedge \varphi$) $\longrightarrow$ ch
$\langle proof \rangle$

lemma MMI˙a1d: assumes A1: $\varphi \longrightarrow \psi$
shows $\varphi \longrightarrow$ (ch $\longrightarrow \psi$)
$\langle proof \rangle$

lemma MMI˙r19˙21aiv: assumes A1: $\forall$x. $\varphi \longrightarrow$ (x $\in$ A $\longrightarrow \psi$(x))
shows $\varphi \longrightarrow$ ($\forall$ x $\in$ A . ψ(x))
$\langle proof \rangle$

lemma MMI˙r19˙22:
shows ($\forall$ x $\in$ A . (φ(x) $\longrightarrow \psi$(x))) $\longrightarrow$
(($\exists$ x $\in$ A . φ(x)) $\longrightarrow$ ($\exists$ x $\in$ A . ψ(x)))
$\langle proof \rangle$

lemma MMI˙syl6: assumes A1: $\varphi \longrightarrow$ ($\psi \longrightarrow$ ch) and
A2: ch $\longrightarrow \vartheta$
shows $\varphi \longrightarrow$ ($\psi \longrightarrow \vartheta$)
$\langle proof \rangle$

lemma MMI˙mpid: assumes A1: $\varphi \longrightarrow$ ch and
A2: $\varphi \longrightarrow$ ($\psi \longrightarrow$ (ch $\longrightarrow \vartheta$))
shows $\varphi \longrightarrow$ ($\psi \longrightarrow \vartheta$)
$\langle proof \rangle$

lemma MMI˙eqtr3t:
shows (A = C $\wedge$ B = C) $\longrightarrow$ A = B

⟨*proof*⟩

lemma MMI˙syl5bi: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow \mathrm{ch})$ and
A2: $\vartheta \longrightarrow \psi$
shows $\varphi \longrightarrow (\vartheta \longrightarrow \mathrm{ch})$
⟨*proof*⟩

lemma MMI˙mp3an1: assumes A1: φ and
A2: $(\varphi \wedge \psi \wedge \mathrm{ch}) \longrightarrow \vartheta$
shows $(\psi \wedge \mathrm{ch}) \longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙rgen2: assumes A1: $\forall x\ y.\ (x \in A \wedge y \in A) \longrightarrow \varphi(x,y)$
shows $\forall\ x \in A\ .\ \forall\ y \in A\ .\ \varphi(x,y)$
⟨*proof*⟩

lemma MMI˙ax˙17: shows $\varphi \longrightarrow (\forall x.\ \varphi)$ ⟨*proof*⟩

lemma MMI˙3eqtr4g: assumes A1: $\varphi \longrightarrow A = B$ and
A2: $C = A$ and
A3: $D = B$
shows $\varphi \longrightarrow C = D$
⟨*proof*⟩

lemma MMI˙3imtr4: assumes A1: $\varphi \longrightarrow \psi$ and
A2: $\mathrm{ch} \longleftrightarrow \varphi$ and
A3: $\vartheta \longleftrightarrow \psi$
shows $\mathrm{ch} \longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙eleq2i: assumes A1: $A = B$
shows $C \in A \longleftrightarrow C \in B$
⟨*proof*⟩

lemma MMI˙albii: assumes A1: $\varphi \longleftrightarrow \psi$
shows $(\forall\ x\ .\ \varphi) \longleftrightarrow (\forall\ x\ .\ \psi)$
⟨*proof*⟩

lemma MMI˙reucl:
shows $(\exists!\ x\ .\ x \in A \wedge \varphi(x)) \longrightarrow \bigcup -x \in A\ .\ \varphi(x)\ '' \in A$
⟨*proof*⟩

lemma MMI˙dedth2h: assumes A1: A = if (φ , A , C) $\longrightarrow$ (ch $\longleftrightarrow$ ϑ) and
A2: B = if (ψ , B , D) $\longrightarrow$ (ϑ $\longleftrightarrow$ τ) and
A3: τ
shows (φ $\wedge$ ψ) $\longrightarrow$ ch
$\langle proof \rangle$

lemma MMI˙eleq1d: assumes A1: φ $\longrightarrow$ A = B
shows φ $\longrightarrow$ (A $\in$ C $\longleftrightarrow$ B $\in$ C)
$\langle proof \rangle$

lemma MMI˙syl5eqel: assumes A1: φ $\longrightarrow$ A $\in$ B and
A2: C = A
shows φ $\longrightarrow$ C $\in$ B
$\langle proof \rangle$

lemma IML˙eeuni: assumes A1: x $\in$ A and A2: $\exists!$ t . t $\in$ A $\wedge$ φ(t)
shows φ(x) $\longleftrightarrow$ $\bigcup$ – x $\in$ A . φ(x) ″ = x
$\langle proof \rangle$

lemma MMI˙reuuni1:
shows (x $\in$ A $\wedge$ ($\exists!$ x . x $\in$ A $\wedge$ φ(x))) $\longrightarrow$
(φ(x) $\longleftrightarrow$ $\bigcup$ – x $\in$ A . φ(x) ″ = x)
$\langle proof \rangle$

lemma MMI˙eqeq1i: assumes A1: A = B
shows A = C $\longleftrightarrow$ B = C
$\langle proof \rangle$

lemma MMI˙syl6rbbr: assumes A1: $\forall$x. φ(x) $\longrightarrow$ (ψ(x) $\longleftrightarrow$ ch(x)) and
A2: $\forall$x. ϑ(x) $\longleftrightarrow$ ch(x)
shows $\forall$ x. φ(x) $\longrightarrow$ (ϑ(x) $\longleftrightarrow$ ψ(x))
$\langle proof \rangle$

lemma MMI˙syl6rbbrA: assumes A1: φ $\longrightarrow$ (ψ $\longleftrightarrow$ ch) and
A2: ϑ $\longleftrightarrow$ ch
shows φ $\longrightarrow$ (ϑ $\longleftrightarrow$ ψ)
$\langle proof \rangle$

lemma MMI˙vtoclga: assumes A1: $\forall$x. x = A $\longrightarrow$ (φ(x) $\longleftrightarrow$ ψ) and
A2: $\forall$x. x $\in$ B $\longrightarrow$ φ(x)
shows A $\in$ B $\longrightarrow$ ψ
$\langle proof \rangle$

lemma MMI˙3bitr4: assumes A1: $\varphi \longleftrightarrow \psi$ and
A2: ch $\longleftrightarrow \varphi$ and
A3: $\vartheta \longleftrightarrow \psi$
shows ch $\longleftrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙mpbii: assumes Amin: ψ and
Amaj: $\varphi \longrightarrow (\psi \longleftrightarrow$ ch)
shows $\varphi \longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙eqid:
shows A = A
⟨*proof*⟩

lemma MMI˙pm3˙27:
shows ($\varphi \wedge \psi$) $\longrightarrow \psi$
⟨*proof*⟩

lemma MMI˙pm3˙26:
shows ($\varphi \wedge \psi$) $\longrightarrow \varphi$
⟨*proof*⟩

lemma MMI˙ancoms: assumes A1: ($\varphi \wedge \psi$) $\longrightarrow$ ch
shows ($\psi \wedge \varphi$) $\longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙syl3anc: assumes A1: ($\varphi \wedge \psi \wedge$ ch) $\longrightarrow \vartheta$ and
A2: $\tau \longrightarrow \varphi$ and
A3: $\tau \longrightarrow \psi$ and
A4: $\tau \longrightarrow$ ch
shows $\tau \longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙syl5eq: assumes A1: $\varphi \longrightarrow$ A = B and
A2: C = A
shows $\varphi \longrightarrow$ C = B
⟨*proof*⟩

lemma MMI˙eqcomi: assumes A1: A = B
shows B = A
⟨*proof*⟩

lemma MMI˙3eqtr: assumes A1: A = B and
A2: B = C and
A3: C = D
shows A = D
⟨*proof*⟩

lemma MMI˙mpbir: assumes Amin: ψ and
Amaj: $\varphi \longleftrightarrow \psi$
shows φ
$\langle proof \rangle$

lemma MMI˙syl3an3: assumes A1: ($\varphi \wedge \psi \wedge$ ch) $\longrightarrow \vartheta$ and
A2: $\tau \longrightarrow$ ch
shows ($\varphi \wedge \psi \wedge \tau$) $\longrightarrow \vartheta$
$\langle proof \rangle$

lemma MMI˙3eqtrd: assumes A1: $\varphi \longrightarrow$ A = B and
A2: $\varphi \longrightarrow$ B = C and
A3: $\varphi \longrightarrow$ C = D
shows $\varphi \longrightarrow$ A = D
$\langle proof \rangle$

lemma MMI˙syl5: assumes A1: $\varphi \longrightarrow$ ($\psi \longrightarrow$ ch) and
A2: $\vartheta \longrightarrow \psi$
shows $\varphi \longrightarrow$ ($\vartheta \longrightarrow$ ch)
$\langle proof \rangle$

lemma MMI˙exp3a: assumes A1: $\varphi \longrightarrow$ (($\psi \wedge$ ch) $\longrightarrow \vartheta$)
shows $\varphi \longrightarrow$ ($\psi \longrightarrow$ (ch $\longrightarrow \vartheta$))
$\langle proof \rangle$

lemma MMI˙com12: assumes A1: $\varphi \longrightarrow$ ($\psi \longrightarrow$ ch)
shows $\psi \longrightarrow$ ($\varphi \longrightarrow$ ch)
$\langle proof \rangle$

lemma MMI˙3imp: assumes A1: $\varphi \longrightarrow$ ($\psi \longrightarrow$ (ch $\longrightarrow \vartheta$))
shows ($\varphi \wedge \psi \wedge$ ch) $\longrightarrow \vartheta$
$\langle proof \rangle$

lemma MMI˙3eqtr3: assumes A1: A = B and
A2: A = C and
A3: B = D
shows C = D
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙opreq1i: assumes A1: A = B
shows
(A + C) = (B + C)
(A - C) = (B - C)
(A / C) = (B / C)
(A $\cdot$ C) = (B $\cdot$ C)
$\langle proof \rangle$

lemma MMI˙eqtr3: assumes A1: A = B and
A2: A = C
shows B = C
⟨*proof*⟩

lemma MMI˙dedth: assumes A1: A = if (φ , A , B) $\longrightarrow$ ($\psi \longleftrightarrow$ ch) and
A2: ch
shows $\varphi \longrightarrow \psi$
⟨*proof*⟩

lemma MMI˙id:
shows $\varphi \longrightarrow \varphi$
⟨*proof*⟩

lemma MMI˙eqtr3d: assumes A1: $\varphi \longrightarrow$ A = B and
A2: $\varphi \longrightarrow$ A = C
shows $\varphi \longrightarrow$ B = C
⟨*proof*⟩

lemma MMI˙sylan2: assumes A1: ($\varphi \wedge \psi$) $\longrightarrow$ ch and
A2: $\vartheta \longrightarrow \psi$
shows ($\varphi \wedge \vartheta$) $\longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙adantl: assumes A1: $\varphi \longrightarrow \psi$
shows (ch $\wedge \varphi$) $\longrightarrow \psi$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙opreq12:
shows
(A = B $\wedge$ C = D) $\longrightarrow$ (A + C) = (B + D)
(A = B $\wedge$ C = D) $\longrightarrow$ (A - C) = (B - D)
(A = B $\wedge$ C = D) $\longrightarrow$ (A $\cdot$ C) = (B $\cdot$ D)
(A = B $\wedge$ C = D) $\longrightarrow$ (A / C) = (B / D)
⟨*proof*⟩

lemma MMI˙anidms: assumes A1: ($\varphi \wedge \varphi$) $\longrightarrow \psi$
shows $\varphi \longrightarrow \psi$
⟨*proof*⟩

lemma MMI˙anabsan2: assumes A1: ($\varphi \wedge$ ($\psi \wedge \psi$)) $\longrightarrow$ ch
shows ($\varphi \wedge \psi$) $\longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙3simp2:
shows ($\varphi \wedge \psi \wedge$ ch) $\longrightarrow \psi$

⟨*proof*⟩

lemma MMI˙3simp3:
shows ($\varphi \wedge \psi \wedge$ ch) $\longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙sylbir: assumes A1: $\psi \longleftrightarrow \varphi$ and
A2: $\psi \longrightarrow$ ch
shows $\varphi \longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙3eqtr3g: assumes A1: $\varphi \longrightarrow$ A = B and
A2: A = C and
A3: B = D
shows $\varphi \longrightarrow$ C = D
⟨*proof*⟩

lemma MMI˙3bitr: assumes A1: $\varphi \longleftrightarrow \psi$ and
A2: $\psi \longleftrightarrow$ ch and
A3: ch $\longleftrightarrow \vartheta$
shows $\varphi \longleftrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙3bitr3: assumes A1: $\varphi \longleftrightarrow \psi$ and
A2: $\varphi \longleftrightarrow$ ch and
A3: $\psi \longleftrightarrow \vartheta$
shows ch $\longleftrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙eqcom:
shows A = B $\longleftrightarrow$ B = A
⟨*proof*⟩

lemma MMI˙syl6bb: assumes A1: $\varphi \longrightarrow$ ($\psi \longleftrightarrow$ ch) and
A2: ch $\longleftrightarrow \vartheta$
shows $\varphi \longrightarrow$ ($\psi \longleftrightarrow \vartheta$)
⟨*proof*⟩

lemma MMI˙3bitr3d: assumes A1: $\varphi \longrightarrow$ ($\psi \longleftrightarrow$ ch) and
A2: $\varphi \longrightarrow$ ($\psi \longleftrightarrow \vartheta$) and
A3: $\varphi \longrightarrow$ (ch $\longleftrightarrow \tau$)
shows $\varphi \longrightarrow$ ($\vartheta \longleftrightarrow \tau$)
⟨*proof*⟩

lemma MMI˙syl3an2: assumes A1: ($\varphi \wedge \psi \wedge$ ch) $\longrightarrow \vartheta$ and
A2: $\tau \longrightarrow \psi$
shows ($\varphi \wedge \tau \wedge$ ch) $\longrightarrow \vartheta$

$\langle proof \rangle$

lemma MMI˙df˙rex:
shows ($\exists$ x $\in$ A . φ(x)) $\longleftrightarrow$ ($\exists$ x . (x $\in$ A $\wedge$ φ(x)))
$\langle proof \rangle$

lemma MMI˙mpbi: assumes Amin: φ and
Amaj: $\varphi \longleftrightarrow \psi$
shows ψ
$\langle proof \rangle$

lemma MMI˙mp3an12: assumes A1: φ and
A2: ψ and
A3: ($\varphi \wedge \psi \wedge$ ch) $\longrightarrow \vartheta$
shows ch $\longrightarrow \vartheta$
$\langle proof \rangle$

lemma MMI˙syl5bb: assumes A1: $\varphi \longrightarrow$ ($\psi \longleftrightarrow$ ch) and
A2: $\vartheta \longleftrightarrow \psi$
shows $\varphi \longrightarrow$ ($\vartheta \longleftrightarrow$ ch)
$\langle proof \rangle$

lemma MMI˙eleq1a:
shows A $\in$ B $\longrightarrow$ (C = A $\longrightarrow$ C $\in$ B)
$\langle proof \rangle$

lemma MMI˙sylbird: assumes A1: $\varphi \longrightarrow$ (ch $\longleftrightarrow \psi$) and
A2: $\varphi \longrightarrow$ (ch $\longrightarrow \vartheta$)
shows $\varphi \longrightarrow$ ($\psi \longrightarrow \vartheta$)
$\langle proof \rangle$

lemma MMI˙19˙23aiv: assumes A1: $\forall$x. φ(x) $\longrightarrow \psi$
shows ($\exists$ x . φ(x)) $\longrightarrow \psi$
$\langle proof \rangle$

lemma MMI˙eqeltrrd: assumes A1: $\varphi \longrightarrow$ A = B and
A2: $\varphi \longrightarrow$ A $\in$ C
shows $\varphi \longrightarrow$ B $\in$ C
$\langle proof \rangle$

lemma MMI˙syl2an: assumes A1: ($\varphi \wedge \psi$) $\longrightarrow$ ch and
A2: $\vartheta \longrightarrow \varphi$ and
A3: $\tau \longrightarrow \psi$
shows ($\vartheta \wedge \tau$) $\longrightarrow$ ch
$\langle proof \rangle$

lemma MMI˙adantrl: assumes A1: ($\varphi \wedge \psi$) $\longrightarrow$ ch
shows ($\varphi \wedge$ ($\vartheta \wedge \psi$)) $\longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙ad2ant2r: assumes A1: ($\varphi \wedge \psi$) $\longrightarrow$ ch
shows (($\varphi \wedge \vartheta$) $\wedge$ ($\psi \wedge \tau$)) $\longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙adantll: assumes A1: ($\varphi \wedge \psi$) $\longrightarrow$ ch
shows (($\vartheta \wedge \varphi$) $\wedge \psi$) $\longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙anandirs: assumes A1: (($\varphi \wedge$ ch) $\wedge$ ($\psi \wedge$ ch)) $\longrightarrow \tau$
shows (($\varphi \wedge \psi$) $\wedge$ ch) $\longrightarrow \tau$
⟨*proof*⟩

lemma MMI˙adantlr: assumes A1: ($\varphi \wedge \psi$) $\longrightarrow$ ch
shows (($\varphi \wedge \vartheta$) $\wedge \psi$) $\longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙an42s: assumes A1: (($\varphi \wedge \psi$) $\wedge$ (ch $\wedge \vartheta$)) $\longrightarrow \tau$
shows (($\varphi \wedge$ ch) $\wedge$ ($\vartheta \wedge \psi$)) $\longrightarrow \tau$
⟨*proof*⟩

lemma MMI˙mp3an2: assumes A1: ψ and
A2: ($\varphi \wedge \psi \wedge$ ch) $\longrightarrow \vartheta$
shows ($\varphi \wedge$ ch) $\longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙3simp1:
shows ($\varphi \wedge \psi \wedge$ ch) $\longrightarrow \varphi$
⟨*proof*⟩

lemma MMI˙3impb: assumes A1: ($\varphi \wedge$ ($\psi \wedge$ ch)) $\longrightarrow \vartheta$
shows ($\varphi \wedge \psi \wedge$ ch) $\longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙mpbird: assumes Amin: $\varphi \longrightarrow$ ch and
Amaj: $\varphi \longrightarrow$ ($\psi \longleftrightarrow$ ch)
shows $\varphi \longrightarrow \psi$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙opreq12i: assumes A1: A = B and
A2: C = D

shows
(A + C) = (B + D)
(A · C) = (B · D)
(A - C) = (B - D)
⟨*proof*⟩

lemma MMI˙3eqtr4: assumes A1: A = B and
A2: C = A and
A3: D = B
shows C = D
⟨*proof*⟩

lemma MMI˙eqtr4d: assumes A1: $\varphi \longrightarrow$ A = B and
A2: $\varphi \longrightarrow$ C = B
shows $\varphi \longrightarrow$ A = C
⟨*proof*⟩

lemma MMI˙3eqtr3rd: assumes A1: $\varphi \longrightarrow$ A = B and
A2: $\varphi \longrightarrow$ A = C and
A3: $\varphi \longrightarrow$ B = D
shows $\varphi \longrightarrow$ D = C
⟨*proof*⟩

lemma MMI˙sylanc: assumes A1: ($\varphi \wedge \psi$) $\longrightarrow$ ch and
A2: $\vartheta \longrightarrow \varphi$ and
A3: $\vartheta \longrightarrow \psi$
shows $\vartheta \longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙anim12i: assumes A1: $\varphi \longrightarrow \psi$ and
A2: ch $\longrightarrow \vartheta$
shows ($\varphi \wedge$ ch) $\longrightarrow$ ($\psi \wedge \vartheta$)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙opreqan12d: assumes A1: $\varphi \longrightarrow$ A = B and
A2: $\psi \longrightarrow$ C = D
shows
($\varphi \wedge \psi$) $\longrightarrow$ (A + C) = (B + D)
($\varphi \wedge \psi$) $\longrightarrow$ (A - C) = (B - D)
($\varphi \wedge \psi$) $\longrightarrow$ (A · C) = (B · D)
⟨*proof*⟩

lemma MMI˙sylanr2: assumes A1: ($\varphi \wedge$ ($\psi \wedge$ ch)) $\longrightarrow \vartheta$ and

A2: $\tau \longrightarrow$ ch
shows ($\varphi \wedge$ ($\psi \wedge \tau$)) $\longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙sylanl2: assumes A1: (($\varphi \wedge \psi$) $\wedge$ ch) $\longrightarrow \vartheta$ and
A2: $\tau \longrightarrow \psi$
shows (($\varphi \wedge \tau$) $\wedge$ ch) $\longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙ancom2s: assumes A1: ($\varphi \wedge$ ($\psi \wedge$ ch)) $\longrightarrow \vartheta$
shows ($\varphi \wedge$ (ch $\wedge \psi$)) $\longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙anandis: assumes A1: (($\varphi \wedge \psi$) $\wedge$ ($\varphi \wedge$ ch)) $\longrightarrow \tau$
shows ($\varphi \wedge$ ($\psi \wedge$ ch)) $\longrightarrow \tau$
⟨*proof*⟩

lemma MMI˙sylan9eq: assumes A1: $\varphi \longrightarrow$ A = B and
A2: $\psi \longrightarrow$ B = C
shows ($\varphi \wedge \psi$) $\longrightarrow$ A = C
⟨*proof*⟩

lemma MMI˙keephyp: assumes A1: A = if (φ , A , B) $\longrightarrow$ ($\psi \longleftrightarrow \vartheta$) and
A2: B = if (φ , A , B) $\longrightarrow$ (ch $\longleftrightarrow \vartheta$) and
A3: ψ and
A4: ch
shows ϑ
⟨*proof*⟩

lemma MMI˙eleq1:
shows A = B $\longrightarrow$ (A $\in$ C $\longleftrightarrow$ B $\in$ C)
⟨*proof*⟩

lemma MMI˙pm4˙2i:
shows $\varphi \longrightarrow$ ($\psi \longleftrightarrow \psi$)
⟨*proof*⟩

lemma MMI˙3anbi123d: assumes A1: $\varphi \longrightarrow$ ($\psi \longleftrightarrow$ ch) and
A2: $\varphi \longrightarrow$ ($\vartheta \longleftrightarrow \tau$) and
A3: $\varphi \longrightarrow$ ($\eta \longleftrightarrow \zeta$)
shows $\varphi \longrightarrow$ (($\psi \wedge \vartheta \wedge \eta$) $\longleftrightarrow$ (ch $\wedge \tau \wedge \zeta$))
⟨*proof*⟩

lemma MMI˙imbi12d: assumes A1: $\varphi \longrightarrow$ ($\psi \longleftrightarrow$ ch) and
A2: $\varphi \longrightarrow$ ($\vartheta \longleftrightarrow \tau$)
shows $\varphi \longrightarrow$ (($\psi \longrightarrow \vartheta$) $\longleftrightarrow$ (ch $\longrightarrow \tau$))
⟨*proof*⟩

lemma MMI˙bitrd: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow \mathrm{ch})$ and
A2: $\varphi \longrightarrow (\mathrm{ch} \longleftrightarrow \vartheta)$
shows $\varphi \longrightarrow (\psi \longleftrightarrow \vartheta)$
⟨*proof*⟩

lemma MMI˙df˙ne:
shows $(A \neq B \longleftrightarrow \neg (A = B))$
⟨*proof*⟩

lemma MMI˙3pm3˙2i: assumes A1: φ and
A2: ψ and
A3: ch
shows $\varphi \wedge \psi \wedge \mathrm{ch}$
⟨*proof*⟩

lemma MMI˙eqeq2i: assumes A1: $A = B$
shows $C = A \longleftrightarrow C = B$
⟨*proof*⟩

lemma MMI˙syl5bbr: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow \mathrm{ch})$ and
A2: $\psi \longleftrightarrow \vartheta$
shows $\varphi \longrightarrow (\vartheta \longleftrightarrow \mathrm{ch})$
⟨*proof*⟩

lemma MMI˙biimpd: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow \mathrm{ch})$
shows $\varphi \longrightarrow (\psi \longrightarrow \mathrm{ch})$
⟨*proof*⟩

lemma MMI˙orrd: assumes A1: $\varphi \longrightarrow (\neg (\psi) \longrightarrow \mathrm{ch})$
shows $\varphi \longrightarrow (\psi \vee \mathrm{ch})$
⟨*proof*⟩

lemma MMI˙jaoi: assumes A1: $\varphi \longrightarrow \psi$ and
A2: $\mathrm{ch} \longrightarrow \psi$
shows $(\varphi \vee \mathrm{ch}) \longrightarrow \psi$
⟨*proof*⟩

lemma MMI˙oridm:
shows $(\varphi \vee \varphi) \longleftrightarrow \varphi$
⟨*proof*⟩

lemma MMI˙orbi1d: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow \mathrm{ch})$
shows $\varphi \longrightarrow ((\psi \vee \vartheta) \longleftrightarrow (\mathrm{ch} \vee \vartheta))$
⟨*proof*⟩

lemma MMI˙orbi2d: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow \mathrm{ch})$
shows $\varphi \longrightarrow ((\vartheta \vee \psi) \longleftrightarrow (\vartheta \vee \mathrm{ch}))$
⟨*proof*⟩

lemma MMI˙3bitr4g: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow \text{ch})$ and
A2: $\vartheta \longleftrightarrow \psi$ and
A3: $\tau \longleftrightarrow \text{ch}$
shows $\varphi \longrightarrow (\vartheta \longleftrightarrow \tau)$
⟨*proof*⟩

lemma MMI˙negbid: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow \text{ch})$
shows $\varphi \longrightarrow (\neg (\psi) \longleftrightarrow \neg (\text{ch}))$
⟨*proof*⟩

lemma MMI˙ioran:
shows $\neg ((\varphi \vee \psi)) \longleftrightarrow$
$(\neg (\varphi) \wedge \neg (\psi))$
⟨*proof*⟩

lemma MMI˙syl6rbb: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow \text{ch})$ and
A2: $\text{ch} \longleftrightarrow \vartheta$
shows $\varphi \longrightarrow (\vartheta \longleftrightarrow \psi)$
⟨*proof*⟩

lemma MMI˙anbi12i: assumes A1: $\varphi \longleftrightarrow \psi$ and
A2: $\text{ch} \longleftrightarrow \vartheta$
shows $(\varphi \wedge \text{ch}) \longleftrightarrow (\psi \wedge \vartheta)$
⟨*proof*⟩

lemma MMI˙keepel: assumes A1: $A \in C$ and
A2: $B \in C$
shows if $(\varphi , A , B) \in C$
⟨*proof*⟩

lemma MMI˙imbi2d: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow \text{ch})$
shows $\varphi \longrightarrow ((\vartheta \longrightarrow \psi) \longleftrightarrow (\vartheta \longrightarrow \text{ch}))$
⟨*proof*⟩

lemma MMI˙eqeltr: assumes $A = B$ and $B \in C$
shows $A \in C$ ⟨*proof*⟩

lemma MMI˙3impia: assumes A1: $(\varphi \wedge \psi) \longrightarrow (\text{ch} \longrightarrow \vartheta)$
shows $(\varphi \wedge \psi \wedge \text{ch}) \longrightarrow \vartheta$

⟨*proof*⟩

lemma MMI˙eqneqd: assumes A1: $\varphi \longrightarrow (A = B \longleftrightarrow C = D)$
shows $\varphi \longrightarrow (A \neq B \longleftrightarrow C \neq D)$
⟨*proof*⟩

lemma MMI˙3ad2ant2: assumes A1: $\varphi \longrightarrow ch$
shows $(\psi \wedge \varphi \wedge \vartheta) \longrightarrow ch$
⟨*proof*⟩

lemma MMI˙mp3anl3: assumes A1: ch and
A2: $((\varphi \wedge \psi \wedge ch) \wedge \vartheta) \longrightarrow \tau$
shows $((\varphi \wedge \psi) \wedge \vartheta) \longrightarrow \tau$
⟨*proof*⟩

lemma MMI˙bitr4d: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow ch)$ and
A2: $\varphi \longrightarrow (\vartheta \longleftrightarrow ch)$
shows $\varphi \longrightarrow (\psi \longleftrightarrow \vartheta)$
⟨*proof*⟩

lemma MMI˙neeq1d: assumes A1: $\varphi \longrightarrow A = B$
shows $\varphi \longrightarrow (A \neq C \longleftrightarrow B \neq C)$
⟨*proof*⟩

lemma MMI˙3anim123i: assumes A1: $\varphi \longrightarrow \psi$ and
A2: $ch \longrightarrow \vartheta$ and
A3: $\tau \longrightarrow \eta$
shows $(\varphi \wedge ch \wedge \tau) \longrightarrow (\psi \wedge \vartheta \wedge \eta)$
⟨*proof*⟩

lemma MMI˙3exp: assumes A1: $(\varphi \wedge \psi \wedge ch) \longrightarrow \vartheta$
shows $\varphi \longrightarrow (\psi \longrightarrow (ch \longrightarrow \vartheta))$
⟨*proof*⟩

lemma MMI˙exp4a: assumes A1: $\varphi \longrightarrow (\psi \longrightarrow ((ch \wedge \vartheta) \longrightarrow \tau))$
shows $\varphi \longrightarrow (\psi \longrightarrow (ch \longrightarrow (\vartheta \longrightarrow \tau)))$
⟨*proof*⟩

lemma MMI˙3imp1: assumes A1: $\varphi \longrightarrow (\psi \longrightarrow (ch \longrightarrow (\vartheta \longrightarrow \tau)))$
shows $((\varphi \wedge \psi \wedge ch) \wedge \vartheta) \longrightarrow \tau$
⟨*proof*⟩

lemma MMI˙anim1i: assumes A1: $\varphi \longrightarrow \psi$
shows $(\varphi \wedge ch) \longrightarrow (\psi \wedge ch)$
⟨*proof*⟩

lemma MMI˙3adantl1: assumes A1: $((\varphi \wedge \psi) \wedge ch) \longrightarrow \vartheta$
shows $((\tau \wedge \varphi \wedge \psi) \wedge ch) \longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙3adantl2: assumes A1: $((\varphi \wedge \psi) \wedge ch) \longrightarrow \vartheta$
shows $((\varphi \wedge \tau \wedge \psi) \wedge ch) \longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙3comr: assumes A1: $(\varphi \wedge \psi \wedge ch) \longrightarrow \vartheta$
shows $(ch \wedge \varphi \wedge \psi) \longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙bitr3: assumes A1: $\psi \longleftrightarrow \varphi$ and
A2: $\psi \longleftrightarrow ch$
shows $\varphi \longleftrightarrow ch$
⟨*proof*⟩

lemma MMI˙anbi12d: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow ch)$ and
A2: $\varphi \longrightarrow (\vartheta \longleftrightarrow \tau)$
shows $\varphi \longrightarrow ((\psi \wedge \vartheta) \longleftrightarrow (ch \wedge \tau))$
⟨*proof*⟩

lemma MMI˙pm3˙26i: assumes A1: $\varphi \wedge \psi$
shows φ
⟨*proof*⟩

lemma MMI˙pm3˙27i: assumes A1: $\varphi \wedge \psi$
shows ψ
⟨*proof*⟩

lemma MMI˙anabsan: assumes A1: $((\varphi \wedge \varphi) \wedge \psi) \longrightarrow ch$
shows $(\varphi \wedge \psi) \longrightarrow ch$
⟨*proof*⟩

lemma MMI˙3eqtr4rd: assumes A1: $\varphi \longrightarrow A = B$ and
A2: $\varphi \longrightarrow C = A$ and
A3: $\varphi \longrightarrow D = B$
shows $\varphi \longrightarrow D = C$
⟨*proof*⟩

lemma MMI˙syl3an1: assumes A1: $(\varphi \wedge \psi \wedge ch) \longrightarrow \vartheta$ and
A2: $\tau \longrightarrow \varphi$
shows $(\tau \wedge \psi \wedge ch) \longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙syl3anl2: assumes A1: (($\varphi \wedge \psi \wedge$ ch) $\wedge \vartheta$) $\longrightarrow \tau$ and
A2: $\eta \longrightarrow \psi$
shows (($\varphi \wedge \eta \wedge$ ch) $\wedge \vartheta$) $\longrightarrow \tau$
⟨*proof*⟩

lemma MMI˙jca: assumes A1: $\varphi \longrightarrow \psi$ and
A2: $\varphi \longrightarrow$ ch
shows $\varphi \longrightarrow$ ($\psi \wedge$ ch)
⟨*proof*⟩

lemma MMI˙3ad2ant3: assumes A1: $\varphi \longrightarrow$ ch
shows ($\psi \wedge \vartheta \wedge \varphi$) $\longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙anim2i: assumes A1: $\varphi \longrightarrow \psi$
shows (ch $\wedge \varphi$) $\longrightarrow$ (ch $\wedge \psi$)
⟨*proof*⟩

lemma MMI˙ancom:
shows ($\varphi \wedge \psi$) $\longleftrightarrow$ ($\psi \wedge \varphi$)
⟨*proof*⟩

lemma MMI˙anbi1i: assumes Aaa: $\varphi \longleftrightarrow \psi$
shows ($\varphi \wedge$ ch) $\longleftrightarrow$ ($\psi \wedge$ ch)
⟨*proof*⟩

lemma MMI˙an42:
shows (($\varphi \wedge \psi$) $\wedge$ (ch $\wedge \vartheta$)) $\longleftrightarrow$
(($\varphi \wedge$ ch) $\wedge$ ($\vartheta \wedge \psi$))
⟨*proof*⟩

lemma MMI˙sylanb: assumes A1: ($\varphi \wedge \psi$) $\longrightarrow$ ch and
A2: $\vartheta \longleftrightarrow \varphi$
shows ($\vartheta \wedge \psi$) $\longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙an4:
shows (($\varphi \wedge \psi$) $\wedge$ (ch $\wedge \vartheta$)) $\longleftrightarrow$
(($\varphi \wedge$ ch) $\wedge$ ($\psi \wedge \vartheta$))
⟨*proof*⟩

lemma MMI˙syl2anb: assumes A1: ($\varphi \wedge \psi$) $\longrightarrow$ ch and
A2: $\vartheta \longleftrightarrow \varphi$ and
A3: $\tau \longleftrightarrow \psi$
shows ($\vartheta \wedge \tau$) $\longrightarrow$ ch

$\langle proof \rangle$

lemma MMI˙eqtr2d: assumes A1: $\varphi \longrightarrow$ A = B and
A2: $\varphi \longrightarrow$ B = C
shows $\varphi \longrightarrow$ C = A
$\langle proof \rangle$

lemma MMI˙sylbid: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow$ ch) and
A2: $\varphi \longrightarrow$ (ch $\longrightarrow \vartheta$)
shows $\varphi \longrightarrow (\psi \longrightarrow \vartheta)$
$\langle proof \rangle$

lemma MMI˙sylanl1: assumes A1: (($\varphi \wedge \psi$) $\wedge$ ch) $\longrightarrow \vartheta$ and
A2: $\tau \longrightarrow \varphi$
shows (($\tau \wedge \psi$) $\wedge$ ch) $\longrightarrow \vartheta$
$\langle proof \rangle$

lemma MMI˙sylan2b: assumes A1: ($\varphi \wedge \psi$) $\longrightarrow$ ch and
A2: $\vartheta \longleftrightarrow \psi$
shows ($\varphi \wedge \vartheta$) $\longrightarrow$ ch
$\langle proof \rangle$

lemma MMI˙pm3˙22:
shows ($\varphi \wedge \psi$) $\longrightarrow$ ($\psi \wedge \varphi$)
$\langle proof \rangle$

lemma MMI˙ancli: assumes A1: $\varphi \longrightarrow \psi$
shows $\varphi \longrightarrow (\varphi \wedge \psi)$
$\langle proof \rangle$

lemma MMI˙ad2antlr: assumes A1: $\varphi \longrightarrow \psi$
shows ((ch $\wedge \varphi$) $\wedge \vartheta$) $\longrightarrow \psi$
$\langle proof \rangle$

lemma MMI˙biimpa: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow$ ch)
shows ($\varphi \wedge \psi$) $\longrightarrow$ ch
$\langle proof \rangle$

lemma MMI˙sylan2i: assumes A1: $\varphi \longrightarrow$ (($\psi \wedge$ ch) $\longrightarrow \vartheta$) and
A2: $\tau \longrightarrow$ ch
shows $\varphi \longrightarrow ((\psi \wedge \tau) \longrightarrow \vartheta)$
$\langle proof \rangle$

lemma MMI˙3jca: assumes A1: $\varphi \longrightarrow \psi$ and
A2: $\varphi \longrightarrow$ ch and
A3: $\varphi \longrightarrow \vartheta$
shows $\varphi \longrightarrow$ ($\psi \wedge$ ch $\wedge \vartheta$)
$\langle proof \rangle$

lemma MMI˙com34: assumes A1: $\varphi \longrightarrow (\psi \longrightarrow (\text{ch} \longrightarrow (\vartheta \longrightarrow \tau)))$
shows $\varphi \longrightarrow (\psi \longrightarrow (\vartheta \longrightarrow (\text{ch} \longrightarrow \tau)))$
⟨*proof*⟩

lemma MMI˙imp43: assumes A1: $\varphi \longrightarrow (\psi \longrightarrow (\text{ch} \longrightarrow (\vartheta \longrightarrow \tau)))$
shows $((\varphi \wedge \psi) \wedge (\text{ch} \wedge \vartheta)) \longrightarrow \tau$
⟨*proof*⟩

lemma MMI˙3anass:
shows $(\varphi \wedge \psi \wedge \text{ch}) \longleftrightarrow (\varphi \wedge (\psi \wedge \text{ch}))$
⟨*proof*⟩

lemma MMI˙3eqtr4r: assumes A1: A = B and
A2: C = A and
A3: D = B
shows D = C
⟨*proof*⟩

lemma MMI˙jctl: assumes A1: ψ
shows $\varphi \longrightarrow (\psi \wedge \varphi)$
⟨*proof*⟩

lemma MMI˙sylibr: assumes A1: $\varphi \longrightarrow \psi$ and
A2: $\text{ch} \longleftrightarrow \psi$
shows $\varphi \longrightarrow \text{ch}$
⟨*proof*⟩

lemma MMI˙mpanl1: assumes A1: φ and
A2: $((\varphi \wedge \psi) \wedge \text{ch}) \longrightarrow \vartheta$
shows $(\psi \wedge \text{ch}) \longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙a1i: assumes A1: φ
shows $\psi \longrightarrow \varphi$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙opreqan12rd: assumes A1: $\varphi \longrightarrow$ A = B and
A2: $\psi \longrightarrow$ C = D
shows
$(\psi \wedge \varphi) \longrightarrow (\text{A} + \text{C}) = (\text{B} + \text{D})$
$(\psi \wedge \varphi) \longrightarrow (\text{A} \cdot \text{C}) = (\text{B} \cdot \text{D})$
$(\psi \wedge \varphi) \longrightarrow (\text{A} - \text{C}) = (\text{B} - \text{D})$
$(\psi \wedge \varphi) \longrightarrow (\text{A} / \text{C}) = (\text{B} / \text{D})$
⟨*proof*⟩

lemma MMI˙3adantl3: assumes A1: $((\varphi \wedge \psi) \wedge \text{ch}) \longrightarrow \vartheta$
shows $((\varphi \wedge \psi \wedge \tau) \wedge \text{ch}) \longrightarrow \vartheta$

⟨*proof*⟩

lemma MMI˙sylbi: assumes A1: $\varphi \longleftrightarrow \psi$ and
A2: $\psi \longrightarrow$ ch
shows $\varphi \longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙eirr:
shows $\neg$ (A $\in$ A)
⟨*proof*⟩

lemma MMI˙eleq1i: assumes A1: A = B
shows A $\in$ C $\longleftrightarrow$ B $\in$ C
⟨*proof*⟩

lemma MMI˙mtbir: assumes A1: $\neg$ (ψ) and
A2: $\varphi \longleftrightarrow \psi$
shows $\neg$ (φ)
⟨*proof*⟩

lemma MMI˙mto: assumes A1: $\neg$ (ψ) and
A2: $\varphi \longrightarrow \psi$
shows $\neg$ (φ)
⟨*proof*⟩

lemma MMI˙df˙nel:
shows (A $\notin$ B $\longleftrightarrow \neg$ (A $\in$ B))
⟨*proof*⟩

lemma MMI˙snid: assumes A1: A isASet
shows A $\in$ – A ″
⟨*proof*⟩

lemma MMI˙en2lp:
shows $\neg$ (A $\in$ B $\wedge$ B $\in$ A)
⟨*proof*⟩

lemma MMI˙imnan:
shows ($\varphi \longrightarrow \neg$ (ψ)) $\longleftrightarrow \neg$ (($\varphi \wedge \psi$))
⟨*proof*⟩

lemma MMI˙sseqtr4: assumes A1: A $\subseteq$ B and
A2: C = B
shows A $\subseteq$ C
⟨*proof*⟩

lemma MMI˙ssun1:
shows A $\subseteq$ (A $\cup$ B)
$\langle proof \rangle$

lemma MMI˙ibar:
shows $\varphi \longrightarrow$ ($\psi \longleftrightarrow$ ($\varphi \wedge \psi$))
$\langle proof \rangle$

lemma MMI˙mtbiri: assumes Amin: $\neg$ (ch) and
Amaj: $\varphi \longrightarrow$ ($\psi \longleftrightarrow$ ch)
shows $\varphi \longrightarrow \neg$ (ψ)
$\langle proof \rangle$

lemma MMI˙con2i: assumes Aa: $\varphi \longrightarrow \neg$ (ψ)
shows $\psi \longrightarrow \neg$ (φ)
$\langle proof \rangle$

lemma MMI˙intnand: assumes A1: $\varphi \longrightarrow \neg$ (ψ)
shows $\varphi \longrightarrow \neg$ ((ch $\wedge \psi$))
$\langle proof \rangle$

lemma MMI˙intnanrd: assumes A1: $\varphi \longrightarrow \neg$ (ψ)
shows $\varphi \longrightarrow \neg$ (($\psi \wedge$ ch))
$\langle proof \rangle$

lemma MMI˙biorf:
shows $\neg$ (φ) $\longrightarrow$ ($\psi \longleftrightarrow$ ($\varphi \vee \psi$))
$\langle proof \rangle$

lemma MMI˙bitr2d: assumes A1: $\varphi \longrightarrow$ ($\psi \longleftrightarrow$ ch) and
A2: $\varphi \longrightarrow$ (ch $\longleftrightarrow \vartheta$)
shows $\varphi \longrightarrow$ ($\vartheta \longleftrightarrow \psi$)
$\langle proof \rangle$

lemma MMI˙orass:
shows (($\varphi \vee \psi$) $\vee$ ch) $\longleftrightarrow$ ($\varphi \vee$ ($\psi \vee$ ch))
$\langle proof \rangle$

lemma MMI˙orcom:
shows ($\varphi \vee \psi$) $\longleftrightarrow$ ($\psi \vee \varphi$)
$\langle proof \rangle$

lemma MMI˙3bitr4d: assumes A1: $\varphi \longrightarrow$ ($\psi \longleftrightarrow$ ch) and
A2: $\varphi \longrightarrow$ ($\vartheta \longleftrightarrow \psi$) and
A3: $\varphi \longrightarrow$ ($\tau \longleftrightarrow$ ch)

shows $\varphi \longrightarrow (\vartheta \longleftrightarrow \tau)$
⟨*proof*⟩

lemma MMI˙3imtr4d: assumes A1: $\varphi \longrightarrow (\psi \longrightarrow \text{ch})$ and
A2: $\varphi \longrightarrow (\vartheta \longleftrightarrow \psi)$ and
A3: $\varphi \longrightarrow (\tau \longleftrightarrow \text{ch})$
shows $\varphi \longrightarrow (\vartheta \longrightarrow \tau)$
⟨*proof*⟩

lemma MMI˙3impdi: assumes A1: $((\varphi \wedge \psi) \wedge (\varphi \wedge \text{ch})) \longrightarrow \vartheta$
shows $(\varphi \wedge \psi \wedge \text{ch}) \longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙bi2anan9: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow \text{ch})$ and
A2: $\vartheta \longrightarrow (\tau \longleftrightarrow \eta)$
shows $(\varphi \wedge \vartheta) \longrightarrow ((\psi \wedge \tau) \longleftrightarrow (\text{ch} \wedge \eta))$
⟨*proof*⟩

lemma MMI˙ssel2:
shows $((A \subseteq B \wedge C \in A) \longrightarrow C \in B)$
⟨*proof*⟩

lemma MMI˙an1rs: assumes A1: $((\varphi \wedge \psi) \wedge \text{ch}) \longrightarrow \vartheta$
shows $((\varphi \wedge \text{ch}) \wedge \psi) \longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙ralbidva: assumes A1: $\forall x. (\varphi \wedge x \in A) \longrightarrow (\psi(x) \longleftrightarrow \text{ch}(x))$
shows $\varphi \longrightarrow ((\forall\ x \in A .\ \psi(x)) \longleftrightarrow (\forall\ x \in A .\ \text{ch}(x)))$
⟨*proof*⟩

lemma MMI˙rexbidva: assumes A1: $\forall x. (\varphi \wedge x \in A) \longrightarrow (\psi(x) \longleftrightarrow \text{ch}(x))$
shows $\varphi \longrightarrow ((\exists\ x \in A .\ \psi(x)) \longleftrightarrow (\exists\ x \in A .\ \text{ch}(x)))$
⟨*proof*⟩

lemma MMI˙con2bid: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow \neg (\text{ch}))$
shows $\varphi \longrightarrow (\text{ch} \longleftrightarrow \neg (\psi))$
⟨*proof*⟩

lemma MMI˙so: assumes
A1: $\forall x\ y\ z. (x \in A \wedge y \in A \wedge z \in A) \longrightarrow$
$((\langle x,y\rangle \in R \longleftrightarrow \neg ((x = y \vee \langle y, x\rangle \in R))) \wedge$
$((\langle x, y\rangle \in R\ \wedge \langle y, z\rangle \in R) \longrightarrow \langle x, z\rangle \in R))$
shows R Orders A
⟨*proof*⟩

lemma MMI˙con1bid: assumes A1: $\varphi \longrightarrow (\neg (\psi) \longleftrightarrow \text{ch})$
shows $\varphi \longrightarrow (\neg (\text{ch}) \longleftrightarrow \psi)$
⟨*proof*⟩

lemma MMI˙sotrieq:
shows ((R Orders A) $\wedge$ (B $\in$ A $\wedge$ C $\in$ A)) $\longrightarrow$
(B = C $\longleftrightarrow$ $\neg$ ((⟨B,C⟩ $\in$ R $\vee$ ⟨C, B⟩ $\in$ R)))
⟨*proof*⟩

lemma MMI˙bicomd: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow \text{ch})$
shows $\varphi \longrightarrow (\text{ch} \longleftrightarrow \psi)$
⟨*proof*⟩

lemma MMI˙sotrieq2:
shows (R Orders A $\wedge$ (B $\in$ A $\wedge$ C $\in$ A)) $\longrightarrow$
(B = C $\longleftrightarrow$ ($\neg$ (⟨B, C⟩ $\in$ R) $\wedge$ $\neg$ (⟨C, B⟩ $\in$ R)))
⟨*proof*⟩

lemma MMI˙orc:
shows $\varphi \longrightarrow (\varphi \vee \psi)$
⟨*proof*⟩

lemma MMI˙syl6bbr: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow \text{ch})$ and
A2: $\vartheta \longleftrightarrow \text{ch}$
shows $\varphi \longrightarrow (\psi \longleftrightarrow \vartheta)$
⟨*proof*⟩

lemma MMI˙orbi1i: assumes A1: $\varphi \longleftrightarrow \psi$
shows $(\varphi \vee \text{ch}) \longleftrightarrow (\psi \vee \text{ch})$
⟨*proof*⟩

lemma MMI˙syl5rbbr: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow \text{ch})$ and
A2: $\psi \longleftrightarrow \vartheta$
shows $\varphi \longrightarrow (\text{ch} \longleftrightarrow \vartheta)$
⟨*proof*⟩

lemma MMI˙anbi2d: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow \text{ch})$
shows $\varphi \longrightarrow ((\vartheta \wedge \psi) \longleftrightarrow (\vartheta \wedge \text{ch}))$
⟨*proof*⟩

lemma MMI˙ord: assumes A1: $\varphi \longrightarrow (\psi \vee \text{ch})$
shows $\varphi \longrightarrow (\neg (\psi) \longrightarrow \text{ch})$
⟨*proof*⟩

lemma MMI˙impbid: assumes A1: $\varphi \longrightarrow (\psi \longrightarrow$ ch) and
A2: $\varphi \longrightarrow ($ ch $\longrightarrow \psi)$
shows $\varphi \longrightarrow (\psi \longleftrightarrow$ ch)
⟨*proof*⟩

lemma MMI˙jcad: assumes A1: $\varphi \longrightarrow (\psi \longrightarrow$ ch) and
A2: $\varphi \longrightarrow (\psi \longrightarrow \vartheta)$
shows $\varphi \longrightarrow (\psi \longrightarrow ($ ch $\wedge \vartheta))$
⟨*proof*⟩

lemma MMI˙ax˙1:
shows $\varphi \longrightarrow (\psi \longrightarrow \varphi)$
⟨*proof*⟩

lemma MMI˙pm2˙24:
shows $\varphi \longrightarrow (\neg (\varphi) \longrightarrow \psi)$
⟨*proof*⟩

lemma MMI˙imp3a: assumes A1: $\varphi \longrightarrow (\psi \longrightarrow ($ ch $\longrightarrow \vartheta))$
shows $\varphi \longrightarrow ((\psi \wedge$ ch $) \longrightarrow \vartheta)$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙breq1:
shows
A = B $\longrightarrow$ (A $\leq$ C $\longleftrightarrow$ B $\leq$ C)
A = B $\longrightarrow$ (A ¡ C $\longleftrightarrow$ B ¡ C)
⟨*proof*⟩

lemma MMI˙biimprd: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow$ ch)
shows $\varphi \longrightarrow ($ ch $\longrightarrow \psi)$
⟨*proof*⟩

lemma MMI˙jaod: assumes A1: $\varphi \longrightarrow (\psi \longrightarrow$ ch) and
A2: $\varphi \longrightarrow (\vartheta \longrightarrow$ ch)
shows $\varphi \longrightarrow ((\psi \vee \vartheta) \longrightarrow$ ch)
⟨*proof*⟩

lemma MMI˙com23: assumes A1: $\varphi \longrightarrow (\psi \longrightarrow ($ ch $\longrightarrow \vartheta))$
shows $\varphi \longrightarrow ($ ch $\longrightarrow (\psi \longrightarrow \vartheta))$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙breq2:
shows
A = B $\longrightarrow$ (C $\leq$ A $\longleftrightarrow$ C $\leq$ B)
A = B $\longrightarrow$ (C ¡ A $\longleftrightarrow$ C ¡ B)
⟨*proof*⟩

lemma MMI˙syld: assumes A1: $\varphi \longrightarrow (\psi \longrightarrow$ ch) and
A2: $\varphi \longrightarrow ($ ch $\longrightarrow \vartheta)$

shows $\varphi \longrightarrow (\psi \longrightarrow \vartheta)$
⟨*proof*⟩

lemma MMI˙biimpcd: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow \text{ch})$
shows $\psi \longrightarrow (\varphi \longrightarrow \text{ch})$
⟨*proof*⟩

lemma MMI˙mp2and: assumes A1: $\varphi \longrightarrow \psi$ and
A2: $\varphi \longrightarrow \text{ch}$ and
A3: $\varphi \longrightarrow ((\psi \wedge \text{ch}) \longrightarrow \vartheta)$
shows $\varphi \longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙sonr:
shows (R Orders A $\wedge$ B $\in$ A) $\longrightarrow \neg$ ($\langle$B,B$\rangle$ $\in$ R)
⟨*proof*⟩

lemma MMI˙orri: assumes A1: $\neg (\varphi) \longrightarrow \psi$
shows $\varphi \vee \psi$
⟨*proof*⟩

lemma MMI˙mpbiri: assumes Amin: ch and
Amaj: $\varphi \longrightarrow (\psi \longleftrightarrow \text{ch})$
shows $\varphi \longrightarrow \psi$
⟨*proof*⟩

lemma MMI˙pm2˙46:
shows $\neg ((\varphi \vee \psi)) \longrightarrow \neg (\psi)$
⟨*proof*⟩

lemma MMI˙elun:
shows A $\in$ (B $\cup$ C) $\longleftrightarrow$ (A $\in$ B $\vee$ A $\in$ C)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙pnfxr:
shows $+\infty \in \mathbb{R}^*$
⟨*proof*⟩

lemma MMI˙elisseti: assumes A1: A $\in$ B
shows A isASet
⟨*proof*⟩

lemma (in MMIsar0) MMI˙mnfxr:
shows $-\infty \in \mathbb{R}^*$
⟨*proof*⟩

lemma MMI˙elpr2: assumes A1: B isASet and

A2: C isASet
shows A $\in$ – B , C ″ $\longleftrightarrow$ (A = B $\vee$ A = C)
⟨*proof*⟩

lemma MMI˙orbi2i: assumes A1: $\varphi \longleftrightarrow \psi$
shows (ch $\vee$ φ) $\longleftrightarrow$ (ch $\vee$ ψ)
⟨*proof*⟩

lemma MMI˙3orass:
shows ($\varphi \vee \psi \vee$ ch) $\longleftrightarrow$ ($\varphi \vee$ ($\psi \vee$ ch))
⟨*proof*⟩

lemma MMI˙bitr4: assumes A1: $\varphi \longleftrightarrow \psi$ and
A2: ch $\longleftrightarrow \psi$
shows $\varphi \longleftrightarrow$ ch
⟨*proof*⟩

lemma MMI˙eleq2:
shows A = B $\longrightarrow$ (C $\in$ A $\longleftrightarrow$ C $\in$ B)
⟨*proof*⟩

lemma MMI˙nelneq:
shows (A $\in$ C $\wedge$ $\neg$ (B $\in$ C)) $\longrightarrow$ $\neg$ (A = B)
⟨*proof*⟩

lemma MMI˙df˙pr:
shows – A , B ″ = (– A ″ $\cup$ – B ″)
⟨*proof*⟩

lemma MMI˙ineq2i: assumes A1: A = B
shows (C $\cap$ A) = (C $\cap$ B)
⟨*proof*⟩

lemma MMI˙mt2: assumes A1: ψ and
A2: $\varphi \longrightarrow \neg$ (ψ)
shows $\neg$ (φ)
⟨*proof*⟩

lemma MMI˙disjsn:
shows (A $\cap$ – B ″) = 0 $\longleftrightarrow$ $\neg$ (B $\in$ A)
⟨*proof*⟩

lemma MMI˙undisj2:
shows ((A $\cap$ B) =
0 $\wedge$ (A $\cap$ C) =
0) $\longleftrightarrow$ (A $\cap$ (B $\cup$ C)) = 0
⟨*proof*⟩

lemma MMI˙disjssun:
shows $((A \cap B) = 0 \longrightarrow (A \subseteq (B \cup C) \longleftrightarrow A \subseteq C))$
⟨*proof*⟩

lemma MMI˙uncom:
shows $(A \cup B) = (B \cup A)$
⟨*proof*⟩

lemma MMI˙sseq2i: assumes A1: $A = B$
shows $(C \subseteq A \longleftrightarrow C \subseteq B)$
⟨*proof*⟩

lemma MMI˙disj:
shows $(A \cap B) =$
$0 \longleftrightarrow (\forall x \in A . \neg (x \in B))$
⟨*proof*⟩

lemma MMI˙syl5ibr: assumes A1: $\varphi \longrightarrow (\psi \longrightarrow ch)$ and
A2: $\psi \longleftrightarrow \vartheta$
shows $\varphi \longrightarrow (\vartheta \longrightarrow ch)$
⟨*proof*⟩

lemma MMI˙con3d: assumes A1: $\varphi \longrightarrow (\psi \longrightarrow ch)$
shows $\varphi \longrightarrow (\neg (ch) \longrightarrow \neg (\psi))$
⟨*proof*⟩

lemma MMI˙dfrex2:
shows $(\exists x \in A . \varphi(x)) \longleftrightarrow \neg ((\forall x \in A . \neg \varphi(x)))$
⟨*proof*⟩

lemma MMI˙visset:
shows x isASet
⟨*proof*⟩

lemma MMI˙elpr: assumes A1: A isASet
shows $A \in – B , C '' \longleftrightarrow (A = B \vee A = C)$
⟨*proof*⟩

lemma MMI˙rexbii: assumes A1: $\forall x. \varphi(x) \longleftrightarrow \psi(x)$
shows $(\exists x \in A . \varphi(x)) \longleftrightarrow (\exists x \in A . \psi(x))$
⟨*proof*⟩

lemma MMI˙r19˙43:
shows $(\exists x \in A . (\varphi(x) \vee \psi(x))) \longleftrightarrow$
$((\exists x \in A . \varphi(x) \vee (\exists x \in A . \psi(x))))$
⟨*proof*⟩

lemma MMI˙exancom:
shows ($\exists$ x . (φ(x) $\wedge$ ψ(x))) $\longleftrightarrow$
($\exists$ x . (ψ(x) $\wedge$ φ(x)))
$\langle proof \rangle$

lemma MMI˙ceqsexv: assumes A1: A isASet and
A2: $\forall$x. x = A $\longrightarrow$ (φ(x) $\longleftrightarrow$ ψ(x))
shows ($\exists$ x . (x = A $\wedge$ φ(x))) $\longleftrightarrow$ ψ(A)
$\langle proof \rangle$

lemma MMI˙orbi12i˙orig: assumes A1: $\varphi \longleftrightarrow \psi$ and
A2: ch $\longleftrightarrow \vartheta$
shows (φ $\vee$ ch) $\longleftrightarrow$ ($\psi \vee \vartheta$)
$\langle proof \rangle$

lemma MMI˙orbi12i: assumes A1: ($\exists$x. φ(x)) $\longleftrightarrow \psi$ and
A2: ($\exists$x. ch(x)) $\longleftrightarrow \vartheta$
shows ($\exists$x. φ(x)) $\vee$ ($\exists$x. ch(x)) $\longleftrightarrow$ ($\psi \vee \vartheta$)
$\langle proof \rangle$

lemma MMI˙syl6ib: assumes A1: $\varphi \longrightarrow$ ($\psi \longrightarrow$ ch) and
A2: ch $\longleftrightarrow \vartheta$
shows $\varphi \longrightarrow$ ($\psi \longrightarrow \vartheta$)
$\langle proof \rangle$

lemma MMI˙intnan: assumes A1: $\neg$ (φ)
shows $\neg$ (($\psi \wedge \varphi$))
$\langle proof \rangle$

lemma MMI˙intnanr: assumes A1: $\neg$ (φ)
shows $\neg$ (($\varphi \wedge \psi$))
$\langle proof \rangle$

lemma MMI˙pm3˙2ni: assumes A1: $\neg$ (φ) and
A2: $\neg$ (ψ)
shows $\neg$ (($\varphi \vee \psi$))
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙breq12:
shows
(A = B $\wedge$ C = D) $\longrightarrow$ (A ¡ C $\longleftrightarrow$ B ¡ D)
(A = B $\wedge$ C = D) $\longrightarrow$ (A $\leq$ C $\longleftrightarrow$ B $\leq$ D)
$\langle proof \rangle$

lemma MMI˙necom:
shows A $\neq$ B $\longleftrightarrow$ B $\neq$ A
$\langle proof \rangle$

lemma MMI˙3jaoi: assumes A1: $\varphi \longrightarrow \psi$ and
A2: ch $\longrightarrow \psi$ and
A3: $\vartheta \longrightarrow \psi$
shows ($\varphi \vee$ ch $\vee \vartheta$) $\longrightarrow \psi$
⟨*proof*⟩

lemma MMI˙jctr: assumes A1: ψ
shows $\varphi \longrightarrow (\varphi \wedge \psi)$
⟨*proof*⟩

lemma MMI˙olc:
shows $\varphi \longrightarrow (\psi \vee \varphi)$
⟨*proof*⟩

lemma MMI˙3syl: assumes A1: $\varphi \longrightarrow \psi$ and
A2: $\psi \longrightarrow$ ch and
A3: ch $\longrightarrow \vartheta$
shows $\varphi \longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙mtbird: assumes Amin: $\varphi \longrightarrow \neg$ (ch) and
Amaj: $\varphi \longrightarrow (\psi \longleftrightarrow$ ch)
shows $\varphi \longrightarrow \neg (\psi)$
⟨*proof*⟩

lemma MMI˙pm2˙21d: assumes A1: $\varphi \longrightarrow \neg (\psi)$
shows $\varphi \longrightarrow (\psi \longrightarrow$ ch)
⟨*proof*⟩

lemma MMI˙3jaodan: assumes A1: ($\varphi \wedge \psi$) $\longrightarrow$ ch and
A2: ($\varphi \wedge \vartheta$) $\longrightarrow$ ch and
A3: ($\varphi \wedge \tau$) $\longrightarrow$ ch
shows ($\varphi \wedge (\psi \vee \vartheta \vee \tau)$) $\longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙sylan2br: assumes A1: ($\varphi \wedge \psi$) $\longrightarrow$ ch and
A2: $\psi \longleftrightarrow \vartheta$
shows ($\varphi \wedge \vartheta$) $\longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙3jaoian: assumes A1: ($\varphi \wedge \psi$) $\longrightarrow$ ch and
A2: ($\vartheta \wedge \psi$) $\longrightarrow$ ch and
A3: ($\tau \wedge \psi$) $\longrightarrow$ ch
shows (($\varphi \vee \vartheta \vee \tau$) $\wedge \psi$) $\longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙mtbid: assumes Amin: $\varphi \longrightarrow \neg (\psi)$ and

Amaj: $\varphi \longrightarrow (\psi \longleftrightarrow \text{ch})$
shows $\varphi \longrightarrow \neg (\text{ch})$
⟨*proof*⟩

lemma MMI˙con1d: assumes A1: $\varphi \longrightarrow (\neg (\psi) \longrightarrow \text{ch})$
shows $\varphi \longrightarrow (\neg (\text{ch}) \longrightarrow \psi)$
⟨*proof*⟩

lemma MMI˙pm2˙21nd: assumes A1: $\varphi \longrightarrow \psi$
shows $\varphi \longrightarrow (\neg (\psi) \longrightarrow \text{ch})$
⟨*proof*⟩

lemma MMI˙syl3an1b: assumes A1: $(\varphi \wedge \psi \wedge \text{ch}) \longrightarrow \vartheta$ and
A2: $\tau \longleftrightarrow \varphi$
shows $(\tau \wedge \psi \wedge \text{ch}) \longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙adantld: assumes A1: $\varphi \longrightarrow (\psi \longrightarrow \text{ch})$
shows $\varphi \longrightarrow ((\vartheta \wedge \psi) \longrightarrow \text{ch})$
⟨*proof*⟩

lemma MMI˙adantrd: assumes A1: $\varphi \longrightarrow (\psi \longrightarrow \text{ch})$
shows $\varphi \longrightarrow ((\psi \wedge \vartheta) \longrightarrow \text{ch})$
⟨*proof*⟩

lemma MMI˙anasss: assumes A1: $((\varphi \wedge \psi) \wedge \text{ch}) \longrightarrow \vartheta$
shows $(\varphi \wedge (\psi \wedge \text{ch})) \longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙syl3an3b: assumes A1: $(\varphi \wedge \psi \wedge \text{ch}) \longrightarrow \vartheta$ and
A2: $\tau \longleftrightarrow \text{ch}$
shows $(\varphi \wedge \psi \wedge \tau) \longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙mpbid: assumes Amin: $\varphi \longrightarrow \psi$ and
Amaj: $\varphi \longrightarrow (\psi \longleftrightarrow \text{ch})$
shows $\varphi \longrightarrow \text{ch}$
⟨*proof*⟩

lemma MMI˙orbi12d: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow \text{ch})$ and
A2: $\varphi \longrightarrow (\vartheta \longleftrightarrow \tau)$
shows $\varphi \longrightarrow ((\psi \vee \vartheta) \longleftrightarrow (\text{ch} \vee \tau))$
⟨*proof*⟩

lemma MMI˙ianor:
shows $\neg (\varphi \wedge \psi) \longleftrightarrow \neg \varphi \vee \neg \psi$

⟨proof⟩

lemma MMI˙bitr2: assumes A1: $\varphi \longleftrightarrow \psi$ and
A2: $\psi \longleftrightarrow$ ch
shows ch $\longleftrightarrow \varphi$
⟨proof⟩

lemma MMI˙biimp: assumes A1: $\varphi \longleftrightarrow \psi$
shows $\varphi \longrightarrow \psi$
⟨proof⟩

lemma MMI˙mpan2d: assumes A1: $\varphi \longrightarrow$ ch and
A2: $\varphi \longrightarrow ((\psi \wedge \text{ch}) \longrightarrow \vartheta)$
shows $\varphi \longrightarrow (\psi \longrightarrow \vartheta)$
⟨proof⟩

lemma MMI˙ad2antrr: assumes A1: $\varphi \longrightarrow \psi$
shows $((\varphi \wedge \text{ch}) \wedge \vartheta) \longrightarrow \psi$
⟨proof⟩

lemma MMI˙biimpac: assumes A1: $\varphi \longrightarrow (\psi \longleftrightarrow \text{ch})$
shows $(\psi \wedge \varphi) \longrightarrow$ ch
⟨proof⟩

lemma MMI˙con2bii: assumes A1: $\varphi \longleftrightarrow \neg (\psi)$
shows $\psi \longleftrightarrow \neg (\varphi)$
⟨proof⟩

lemma MMI˙pm3˙26bd: assumes A1: $\varphi \longleftrightarrow (\psi \wedge \text{ch})$
shows $\varphi \longrightarrow \psi$
⟨proof⟩

lemma MMI˙biimpr: assumes A1: $\varphi \longleftrightarrow \psi$
shows $\psi \longrightarrow \varphi$
⟨proof⟩

lemma (in MMIsar0) MMI˙3brtr3g: assumes A1: $\varphi \longrightarrow$ A ¡ B and
A2: A = C and
A3: B = D
shows $\varphi \longrightarrow$ C ¡ D
⟨proof⟩

lemma (in MMIsar0) MMI˙breq12i: assumes A1: A = B and
A2: C = D
shows

A ¡ C $\longleftrightarrow$ B ¡ D
A $\leq$ C $\longleftrightarrow$ B $\leq$ D
$\langle proof \rangle$

lemma MMI˙negbii: assumes Aa: $\varphi \longleftrightarrow \psi$
shows $\neg\varphi \longleftrightarrow \neg\psi$
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙breq1i: assumes A1: A = B
shows
A ¡ C $\longleftrightarrow$ B ¡ C
A $\leq$ C $\longleftrightarrow$ B $\leq$ C
$\langle proof \rangle$

lemma MMI˙syl5eqr: assumes A1: $\varphi \longrightarrow$ A = B and
A2: A = C
shows $\varphi \longrightarrow$ C = B
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙breq2d: assumes A1: $\varphi \longrightarrow$ A = B
shows
$\varphi \longrightarrow$ C ¡ A $\longleftrightarrow$ C ¡ B
$\varphi \longrightarrow$ C $\leq$ A $\longleftrightarrow$ C $\leq$ B
$\langle proof \rangle$

lemma MMI˙ccase: assumes A1: $\varphi \wedge \psi \longrightarrow \tau$ and
A2: ch $\wedge\ \psi \longrightarrow \tau$ and
A3: $\varphi \wedge \vartheta \longrightarrow \tau$ and
A4: ch $\wedge\ \vartheta \longrightarrow \tau$
shows ($\varphi \vee$ ch) $\wedge$ ($\psi \vee \vartheta$) $\longrightarrow \tau$
$\langle proof \rangle$

lemma MMI˙pm3˙27bd: assumes A1: $\varphi \longleftrightarrow \psi \wedge$ ch
shows $\varphi \longrightarrow$ ch
$\langle proof \rangle$

lemma MMI˙nsyl3: assumes A1: $\varphi \longrightarrow \neg\psi$ and
A2: ch $\longrightarrow \psi$
shows ch $\longrightarrow \neg\varphi$
$\langle proof \rangle$

lemma MMI˙jctild: assumes A1: $\varphi \longrightarrow \psi \longrightarrow$ ch and
A2: $\varphi \longrightarrow \vartheta$
shows $\varphi \longrightarrow$
$\psi \longrightarrow \vartheta \wedge$ ch

⟨*proof*⟩

lemma MMI˙jctird: assumes A1: $\varphi \longrightarrow \psi \longrightarrow$ ch and
A2: $\varphi \longrightarrow \vartheta$
shows $\varphi \longrightarrow$
$\psi \longrightarrow$ ch $\wedge\ \vartheta$
⟨*proof*⟩

lemma MMI˙ccase2: assumes A1: $\varphi \wedge \psi \longrightarrow \tau$ and
A2: ch $\longrightarrow \tau$ and
A3: $\vartheta \longrightarrow \tau$
shows ($\varphi \vee$ ch) $\wedge$ ($\psi \vee \vartheta$) $\longrightarrow \tau$
⟨*proof*⟩

lemma MMI˙3bitr3r: assumes A1: $\varphi \longleftrightarrow \psi$ and
A2: $\varphi \longleftrightarrow$ ch and
A3: $\psi \longleftrightarrow \vartheta$
shows $\vartheta \longleftrightarrow$ ch
⟨*proof*⟩

lemma (in MMIsar0) MMI˙syl6breq: assumes A1: $\varphi \longrightarrow$ A ¡ B and
A2: B = C
shows
$\varphi \longrightarrow$ A ¡ C
⟨*proof*⟩

lemma MMI˙pm2˙61i: assumes A1: $\varphi \longrightarrow \psi$ and
A2: $\neg\varphi \longrightarrow \psi$
shows ψ
⟨*proof*⟩

lemma MMI˙syl6req: assumes A1: $\varphi \longrightarrow$ A = B and
A2: B = C
shows $\varphi \longrightarrow$ C = A
⟨*proof*⟩

lemma MMI˙pm2˙61d: assumes A1: $\varphi \longrightarrow \psi \longrightarrow$ ch and
A2: $\varphi \longrightarrow$
$\neg\psi \longrightarrow$ ch
shows $\varphi \longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙orim1d: assumes A1: $\varphi \longrightarrow \psi \longrightarrow$ ch
shows $\varphi \longrightarrow$
$\psi \vee \vartheta \longrightarrow$ ch $\vee\ \vartheta$

⟨*proof*⟩

lemma (in MMIsar0) MMI˙breq1d: assumes A1: $\varphi \longrightarrow$ A = B
shows
$\varphi \longrightarrow$ A ¡ C $\longleftrightarrow$ B ¡ C
$\varphi \longrightarrow$ A $\leq$ C $\longleftrightarrow$ B $\leq$ C
⟨*proof*⟩

lemma (in MMIsar0) MMI˙breq12d: assumes A1: $\varphi \longrightarrow$ A = B and
A2: $\varphi \longrightarrow$ C = D
shows
$\varphi \longrightarrow$ A ¡ C $\longleftrightarrow$ B ¡ D
$\varphi \longrightarrow$ A $\leq$ C $\longleftrightarrow$ B $\leq$ D
⟨*proof*⟩

lemma MMI˙bibi2d: assumes A1: $\varphi \longrightarrow$
$\psi \longleftrightarrow$ ch
shows $\varphi \longrightarrow$
($\vartheta \longleftrightarrow \psi$) $\longleftrightarrow$
$\vartheta \longleftrightarrow$ ch
⟨*proof*⟩

lemma MMI˙con4bid: assumes A1: $\varphi \longrightarrow$
$\neg\psi \longleftrightarrow \neg$ch
shows $\varphi \longrightarrow$
$\psi \longleftrightarrow$ ch
⟨*proof*⟩

lemma MMI˙3com13: assumes A1: $\varphi \wedge \psi \wedge$ ch $\longrightarrow \vartheta$
shows ch $\wedge\ \psi \wedge \varphi \longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙3bitr3rd: assumes A1: $\varphi \longrightarrow$
$\psi \longleftrightarrow$ ch and
A2: $\varphi \longrightarrow$
$\psi \longleftrightarrow \vartheta$ and
A3: $\varphi \longrightarrow$
ch $\longleftrightarrow \tau$
shows $\varphi \longrightarrow$
$\tau \longleftrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙3imtr4g: assumes A1: $\varphi \longrightarrow \psi \longrightarrow$ ch and
A2: $\vartheta \longleftrightarrow \psi$ and
A3: $\tau \longleftrightarrow$ ch
shows $\varphi \longrightarrow$

$\vartheta \longrightarrow \tau$
$\langle proof \rangle$

lemma MMI˙expcom: assumes A1: $\varphi \wedge \psi \longrightarrow$ ch
shows $\psi \longrightarrow \varphi \longrightarrow$ ch
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙breq2i: assumes A1: A = B
shows
C ¡ A $\longleftrightarrow$ C ¡ B
C $\leq$ A $\longleftrightarrow$ C $\leq$ B
$\langle proof \rangle$

lemma MMI˙3bitr2r: assumes A1: $\varphi \longleftrightarrow \psi$ and
A2: ch $\longleftrightarrow \psi$ and
A3: ch $\longleftrightarrow \vartheta$
shows $\vartheta \longleftrightarrow \varphi$
$\langle proof \rangle$

lemma MMI˙dedth4h: assumes A1: A = if(φ, A, R) $\longrightarrow$
$\tau \longleftrightarrow \eta$ and
A2: B = if(ψ, B, S) $\longrightarrow$
$\eta \longleftrightarrow \zeta$ and
A3: C = if(ch, C, F) $\longrightarrow$
$\zeta \longleftrightarrow$ si and
A4: D = if(ϑ, D, G) $\longrightarrow$ si $\longleftrightarrow$ rh and
A5: rh
shows $(\varphi \wedge \psi) \wedge$ ch $\wedge\ \vartheta \longrightarrow \tau$
$\langle proof \rangle$

lemma MMI˙anbi1d: assumes A1: $\varphi \longrightarrow$
$\psi \longleftrightarrow$ ch
shows $\varphi \longrightarrow$
$\psi \wedge \vartheta \longleftrightarrow$ ch $\wedge\ \vartheta$
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙breqtrrd: assumes A1: $\varphi \longrightarrow$ A ¡ B and
A2: $\varphi \longrightarrow$ C = B
shows $\varphi \longrightarrow$ A ¡ C
$\langle proof \rangle$

lemma MMI˙syl3an: assumes A1: $\varphi \wedge \psi \wedge$ ch $\longrightarrow \vartheta$ and
A2: $\tau \longrightarrow \varphi$ and
A3: $\eta \longrightarrow \psi$ and

A4: $\zeta \longrightarrow$ ch
shows $\tau \wedge \eta \wedge \zeta \longrightarrow \vartheta$
$\langle proof \rangle$

lemma MMI˙3bitrd: assumes A1: $\varphi \longrightarrow$
$\psi \longleftrightarrow$ ch and
A2: $\varphi \longrightarrow$
ch $\longleftrightarrow \vartheta$ and
A3: $\varphi \longrightarrow$
$\vartheta \longleftrightarrow \tau$
shows $\varphi \longrightarrow$
$\psi \longleftrightarrow \tau$
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙breqtr: assumes A1: A ¡ B and
A2: B = C
shows A ¡ C
$\langle proof \rangle$

lemma MMI˙mpi: assumes A1: ψ and
A2: $\varphi \longrightarrow \psi \longrightarrow$ ch
shows $\varphi \longrightarrow$ ch
$\langle proof \rangle$

lemma MMI˙eqtr2: assumes A1: A = B and
A2: B = C
shows C = A
$\langle proof \rangle$

lemma MMI˙eqneqi: assumes A1: A = B $\longleftrightarrow$ C = D
shows A $\neq$ B $\longleftrightarrow$ C $\neq$ D
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙eqbrtrrd: assumes A1: $\varphi \longrightarrow$ A = B and
A2: $\varphi \longrightarrow$ A ¡ C
shows $\varphi \longrightarrow$ B ¡ C
$\langle proof \rangle$

lemma MMI˙mpd: assumes A1: $\varphi \longrightarrow \psi$ and
A2: $\varphi \longrightarrow \psi \longrightarrow$ ch
shows $\varphi \longrightarrow$ ch
$\langle proof \rangle$

lemma MMI˙mpdan: assumes A1: $\varphi \longrightarrow \psi$ and
A2: $\varphi \wedge \psi \longrightarrow$ ch

shows $\varphi \longrightarrow$ ch
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙breqtrd: assumes A1: $\varphi \longrightarrow$ A ¡ B and
A2: $\varphi \longrightarrow$ B = C
shows $\varphi \longrightarrow$ A ¡ C
$\langle proof \rangle$

lemma MMI˙mpand: assumes A1: $\varphi \longrightarrow \psi$ and
A2: $\varphi \longrightarrow$
$\psi \wedge$ ch $\longrightarrow \vartheta$
shows $\varphi \longrightarrow$ ch $\longrightarrow \vartheta$
$\langle proof \rangle$

lemma MMI˙imbi1d: assumes A1: $\varphi \longrightarrow$
$\psi \longleftrightarrow$ ch
shows $\varphi \longrightarrow$
$(\psi \longrightarrow \vartheta) \longleftrightarrow$
(ch $\longrightarrow \vartheta$)
$\langle proof \rangle$

lemma MMI˙mtbii: assumes Amin: $\neg\psi$ and
Amaj: $\varphi \longrightarrow$
$\psi \longleftrightarrow$ ch
shows $\varphi \longrightarrow \neg$ch
$\langle proof \rangle$

lemma MMI˙sylan2d: assumes A1: $\varphi \longrightarrow$
$\psi \wedge$ ch $\longrightarrow \vartheta$ and
A2: $\varphi \longrightarrow \tau \longrightarrow$ ch
shows $\varphi \longrightarrow$
$\psi \wedge \tau \longrightarrow \vartheta$
$\langle proof \rangle$

lemma MMI˙imp32: assumes A1: $\varphi \longrightarrow$
$\psi \longrightarrow$ ch $\longrightarrow \vartheta$
shows $\varphi \wedge \psi \wedge$ ch $\longrightarrow \vartheta$
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙breqan12d: assumes A1: $\varphi \longrightarrow$ A = B and
A2: $\psi \longrightarrow$ C = D
shows
$\varphi \wedge \psi \longrightarrow$ A ¡ C $\longleftrightarrow$ B ¡ D
$\varphi \wedge \psi \longrightarrow$ A $\leq$ C $\longleftrightarrow$ B $\leq$ D

$\langle proof \rangle$

lemma MMI˙a1dd: assumes A1: $\varphi \longrightarrow \psi \longrightarrow$ ch
shows $\varphi \longrightarrow$
$\psi \longrightarrow \vartheta \longrightarrow$ ch
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙3brtr3d: assumes A1: $\varphi \longrightarrow$ A $\leq$ B and
A2: $\varphi \longrightarrow$ A = C and
A3: $\varphi \longrightarrow$ B = D
shows $\varphi \longrightarrow$ C $\leq$ D
$\langle proof \rangle$

lemma MMI˙ad2antll: assumes A1: $\varphi \longrightarrow \psi$
shows ch $\wedge \vartheta \wedge \varphi \longrightarrow \psi$
$\langle proof \rangle$

lemma MMI˙adantrrl: assumes A1: $\varphi \wedge \psi \wedge$ ch $\longrightarrow \vartheta$
shows $\varphi \wedge \psi \wedge \tau \wedge$ ch $\longrightarrow \vartheta$
$\langle proof \rangle$

lemma MMI˙syl2ani: assumes A1: $\varphi \longrightarrow$
$\psi \wedge$ ch $\longrightarrow \vartheta$ and
A2: $\tau \longrightarrow \psi$ and
A3: $\eta \longrightarrow$ ch
shows $\varphi \longrightarrow$
$\tau \wedge \eta \longrightarrow \vartheta$
$\langle proof \rangle$

lemma MMI˙im2anan9: assumes A1: $\varphi \longrightarrow \psi \longrightarrow$ ch and
A2: $\vartheta \longrightarrow$
$\tau \longrightarrow \eta$
shows $\varphi \wedge \vartheta \longrightarrow$
$\psi \wedge \tau \longrightarrow$ ch $\wedge \eta$
$\langle proof \rangle$

lemma MMI˙ancomsd: assumes A1: $\varphi \longrightarrow$
$\psi \wedge$ ch $\longrightarrow \vartheta$
shows $\varphi \longrightarrow$
ch $\wedge \psi \longrightarrow \vartheta$
$\langle proof \rangle$

lemma MMI˙mpani: assumes A1: ψ and
A2: $\varphi \longrightarrow$
$\psi \wedge$ ch $\longrightarrow \vartheta$
shows $\varphi \longrightarrow$ ch $\longrightarrow \vartheta$
$\langle proof \rangle$

lemma MMI˙syldan: assumes A1: $\varphi \wedge \psi \longrightarrow$ ch and

A2: $\varphi \wedge$ ch $\longrightarrow \vartheta$
shows $\varphi \wedge \psi \longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙mp3anl1: assumes A1: φ and
A2: $(\varphi \wedge \psi \wedge$ ch$) \wedge \vartheta \longrightarrow \tau$
shows $(\psi \wedge$ ch$) \wedge \vartheta \longrightarrow \tau$
⟨*proof*⟩

lemma MMI˙3ad2ant1: assumes A1: $\varphi \longrightarrow$ ch
shows $\varphi \wedge \psi \wedge \vartheta \longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙pm3˙2:
shows $\varphi \longrightarrow$
$\psi \longrightarrow \varphi \wedge \psi$
⟨*proof*⟩

lemma MMI˙pm2˙43i: assumes A1: $\varphi \longrightarrow$
$\varphi \longrightarrow \psi$
shows $\varphi \longrightarrow \psi$
⟨*proof*⟩

lemma MMI˙jctil: assumes A1: $\varphi \longrightarrow \psi$ and
A2: ch
shows $\varphi \longrightarrow$ ch $\wedge \psi$
⟨*proof*⟩

lemma MMI˙mpanl12: assumes A1: φ and
A2: ψ and
A3: $(\varphi \wedge \psi) \wedge$ ch $\longrightarrow \vartheta$
shows ch $\longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙mpanr1: assumes A1: ψ and
A2: $\varphi \wedge \psi \wedge$ ch $\longrightarrow \vartheta$
shows $\varphi \wedge$ ch $\longrightarrow \vartheta$
⟨*proof*⟩

lemma MMI˙ad2antrl: assumes A1: $\varphi \longrightarrow \psi$
shows ch $\wedge \varphi \wedge \vartheta \longrightarrow \psi$
⟨*proof*⟩

lemma MMI˙3adant3r: assumes A1: $\varphi \wedge \psi \wedge$ ch $\longrightarrow \vartheta$
shows $\varphi \wedge \psi \wedge$ ch $\wedge \tau \longrightarrow \vartheta$

$\langle proof \rangle$

lemma MMI˙3adant1l: assumes A1: $\varphi \wedge \psi \wedge \text{ch} \longrightarrow \vartheta$
shows $(\tau \wedge \varphi) \wedge \psi \wedge \text{ch} \longrightarrow \vartheta$
$\langle proof \rangle$

lemma MMI˙3adant2r: assumes A1: $\varphi \wedge \psi \wedge \text{ch} \longrightarrow \vartheta$
shows $\varphi \wedge (\psi \wedge \tau) \wedge \text{ch} \longrightarrow \vartheta$
$\langle proof \rangle$

lemma MMI˙3bitr4rd: assumes A1: $\varphi \longrightarrow$
$\psi \longleftrightarrow \text{ch}$ and
A2: $\varphi \longrightarrow$
$\vartheta \longleftrightarrow \psi$ and
A3: $\varphi \longrightarrow$
$\tau \longleftrightarrow \text{ch}$
shows $\varphi \longrightarrow$
$\tau \longleftrightarrow \vartheta$
$\langle proof \rangle$

lemma MMI˙3anrev:
shows $\varphi \wedge \psi \wedge \text{ch} \longleftrightarrow \text{ch} \wedge \psi \wedge \varphi$
$\langle proof \rangle$

lemma MMI˙eqtr4: assumes A1: A = B and
A2: C = B
shows A = C
$\langle proof \rangle$

lemma MMI˙anidm:
shows $\varphi \wedge \varphi \longleftrightarrow \varphi$
$\langle proof \rangle$

lemma MMI˙bi2anan9r: assumes A1: $\varphi \longrightarrow$
$\psi \longleftrightarrow \text{ch}$ and
A2: $\vartheta \longrightarrow$
$\tau \longleftrightarrow \eta$
shows $\vartheta \wedge \varphi \longrightarrow$
$\psi \wedge \tau \longleftrightarrow \text{ch} \wedge \eta$
$\langle proof \rangle$

lemma MMI˙3imtr3g: assumes A1: $\varphi \longrightarrow \psi \longrightarrow \text{ch}$ and
A2: $\psi \longleftrightarrow \vartheta$ and
A3: $\text{ch} \longleftrightarrow \tau$
shows $\varphi \longrightarrow$
$\vartheta \longrightarrow \tau$
$\langle proof \rangle$

lemma MMI˙a3d: assumes A1: $\varphi \longrightarrow$
$\neg\psi \longrightarrow \neg$ch
shows $\varphi \longrightarrow$ ch $\longrightarrow \psi$
⟨*proof*⟩

lemma MMI˙sylan9bbr: assumes A1: $\varphi \longrightarrow$
$\psi \longleftrightarrow$ ch and
A2: $\vartheta \longrightarrow$
ch $\longleftrightarrow \tau$
shows $\vartheta \wedge \varphi \longrightarrow$
$\psi \longleftrightarrow \tau$
⟨*proof*⟩

lemma MMI˙sylan9bb: assumes A1: $\varphi \longrightarrow$
$\psi \longleftrightarrow$ ch and
A2: $\vartheta \longrightarrow$
ch $\longleftrightarrow \tau$
shows $\varphi \wedge \vartheta \longrightarrow$
$\psi \longleftrightarrow \tau$
⟨*proof*⟩

lemma MMI˙3bitr3g: assumes A1: $\varphi \longrightarrow$
$\psi \longleftrightarrow$ ch and
A2: $\psi \longleftrightarrow \vartheta$ and
A3: ch $\longleftrightarrow \tau$
shows $\varphi \longrightarrow$
$\vartheta \longleftrightarrow \tau$
⟨*proof*⟩

lemma MMI˙pm5˙21:
shows $\neg\varphi \wedge \neg\psi \longrightarrow$
$\varphi \longleftrightarrow \psi$
⟨*proof*⟩

lemma MMI˙an6:
shows $(\varphi \wedge \psi \wedge$ ch$) \wedge \vartheta \wedge \tau \wedge \eta \longleftrightarrow$
$(\varphi \wedge \vartheta) \wedge (\psi \wedge \tau) \wedge$ ch $\wedge\, \eta$
⟨*proof*⟩

lemma MMI˙syl3anl1: assumes A1: $(\varphi \wedge \psi \wedge$ ch$) \wedge \vartheta \longrightarrow \tau$ and
A2: $\eta \longrightarrow \varphi$
shows $(\eta \wedge \psi \wedge$ ch$) \wedge \vartheta \longrightarrow \tau$
⟨*proof*⟩

lemma MMI˙imp4a: assumes A1: $\varphi \longrightarrow$
$\psi \longrightarrow$

ch $\longrightarrow$
$\vartheta \longrightarrow \tau$
shows $\varphi \longrightarrow$
$\psi \longrightarrow$
ch $\wedge\ \vartheta \longrightarrow \tau$
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙breqan12rd: assumes A1: $\varphi \longrightarrow$ A = B and
A2: $\psi \longrightarrow$ C = D
shows
$\psi \wedge \varphi \longrightarrow$ A ¡ C $\longleftrightarrow$ B ¡ D
$\psi \wedge \varphi \longrightarrow$ A $\leq$ C $\longleftrightarrow$ B $\leq$ D
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙3brtr4d: assumes A1: $\varphi \longrightarrow$ A ¡ B and
A2: $\varphi \longrightarrow$ C = A and
A3: $\varphi \longrightarrow$ D = B
shows $\varphi \longrightarrow$ C ¡ D
$\langle proof \rangle$

lemma MMI˙adantrrr: assumes A1: $\varphi \wedge \psi \wedge$ ch $\longrightarrow \vartheta$
shows $\varphi \wedge \psi \wedge$ ch $\wedge\ \tau \longrightarrow \vartheta$
$\langle proof \rangle$

lemma MMI˙adantrlr: assumes A1: $\varphi \wedge \psi \wedge$ ch $\longrightarrow \vartheta$
shows $\varphi \wedge (\psi \wedge \tau) \wedge$ ch $\longrightarrow \vartheta$
$\langle proof \rangle$

lemma MMI˙imdistani: assumes A1: $\varphi \longrightarrow \psi \longrightarrow$ ch
shows $\varphi \wedge \psi \longrightarrow \varphi \wedge$ ch
$\langle proof \rangle$

lemma MMI˙anabss3: assumes A1: $(\varphi \wedge \psi) \wedge \psi \longrightarrow$ ch
shows $\varphi \wedge \psi \longrightarrow$ ch
$\langle proof \rangle$

lemma MMI˙mp3anl2: assumes A1: ψ and
A2: $(\varphi \wedge \psi \wedge$ ch$) \wedge \vartheta \longrightarrow \tau$
shows $(\varphi \wedge$ ch$) \wedge \vartheta \longrightarrow \tau$
$\langle proof \rangle$

lemma MMI˙mpanl2: assumes A1: ψ and
A2: $(\varphi \wedge \psi) \wedge$ ch $\longrightarrow \vartheta$
shows $\varphi \wedge$ ch $\longrightarrow \vartheta$
$\langle proof \rangle$

lemma MMI˙mpancom: assumes A1: $\psi \longrightarrow \varphi$ and
A2: $\varphi \wedge \psi \longrightarrow$ ch
shows $\psi \longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙or12:
shows $\varphi \vee \psi \vee$ ch $\longleftrightarrow \psi \vee \varphi \vee$ ch
⟨*proof*⟩

lemma MMI˙rcla4ev: assumes A1: $\forall$x. x = A $\longrightarrow$ φ(x) $\longleftrightarrow \psi$
shows A $\in$ B $\wedge \psi \longrightarrow$ ($\exists$x$\in$B. φ(x))
⟨*proof*⟩

lemma MMI˙jctir: assumes A1: $\varphi \longrightarrow \psi$ and
A2: ch
shows $\varphi \longrightarrow \psi \wedge$ ch
⟨*proof*⟩

lemma MMI˙iffalse:
shows $\neg\varphi \longrightarrow$ if(φ, A, B) = B
⟨*proof*⟩

lemma MMI˙iftrue:
shows $\varphi \longrightarrow$ if(φ, A, B) = A
⟨*proof*⟩

lemma MMI˙pm2˙61d2: assumes A1: $\varphi \longrightarrow$
$\neg\psi \longrightarrow$ ch and
A2: $\psi \longrightarrow$ ch
shows $\varphi \longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙pm2˙61dan: assumes A1: $\varphi \wedge \psi \longrightarrow$ ch and
A2: $\varphi \wedge \neg\psi \longrightarrow$ ch
shows $\varphi \longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙orcanai: assumes A1: $\varphi \longrightarrow \psi \vee$ ch
shows $\varphi \wedge \neg\psi \longrightarrow$ ch
⟨*proof*⟩

lemma MMI˙ifcl:
shows A $\in$ C $\wedge$ B $\in$ C $\longrightarrow$ if(φ, A, B) $\in$ C
⟨*proof*⟩

lemma MMI˙imim2i: assumes A1: $\varphi \longrightarrow \psi$
shows (ch $\longrightarrow \varphi$) $\longrightarrow$ ch $\longrightarrow \psi$
⟨*proof*⟩

lemma MMI˙com13: assumes A1: $\varphi \longrightarrow$
$\psi \longrightarrow$ ch $\longrightarrow \vartheta$
shows ch $\longrightarrow$
$\psi \longrightarrow$
$\varphi \longrightarrow \vartheta$
$\langle proof \rangle$

lemma MMI˙rcla4v: assumes A1: $\forall$x. x = A $\longrightarrow$ φ(x) $\longleftrightarrow \psi$
shows A $\in$ B $\longrightarrow$ ($\forall$x$\in$B. φ(x)) $\longrightarrow \psi$
$\langle proof \rangle$

lemma MMI˙syl5d: assumes A1: $\varphi \longrightarrow$
$\psi \longrightarrow$ ch $\longrightarrow \vartheta$ and
A2: $\varphi \longrightarrow \tau \longrightarrow$ ch
shows $\varphi \longrightarrow$
$\psi \longrightarrow$
$\tau \longrightarrow \vartheta$
$\langle proof \rangle$

lemma MMI˙eqcoms: assumes A1: A = B $\longrightarrow \varphi$
shows B = A $\longrightarrow \varphi$
$\langle proof \rangle$

lemma MMI˙rgen: assumes A1: $\forall$x. x $\in$ A $\longrightarrow \varphi$(x)
shows $\forall$x$\in$A. φ(x)
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙reex:
shows $\mathbb{R} = \mathbb{R}$
$\langle proof \rangle$

lemma MMI˙sstri: assumes A1: A $\subseteq$B and
A2: B $\subseteq$C
shows A $\subseteq$C
$\langle proof \rangle$

lemma MMI˙ssexi: assumes A1: B = B and
A2: A $\subseteq$B
shows A = A
$\langle proof \rangle$

end

79 Complex numbers in Metamatah - introduction

theory MMI_Complex_ZF imports MMI_logic_and_sets

begin

This theory contains theorems (with proofs) about complex numbers imported from the Metamath's set.mm database. The original Metamath proofs were mostly written by Norman Megill, see the Metamath Proof Explorer pages for full atribution. This theory contains about 200 theorems from "recnt" to "div11t".

lemma (in MMIsar0) MMI_recnt:
  shows $A \in \mathbb{R} \longrightarrow A \in \mathbb{C}$
$\langle proof \rangle$

lemma (in MMIsar0) MMI_recn: assumes A1: $A \in \mathbb{R}$
  shows $A \in \mathbb{C}$
$\langle proof \rangle$

lemma (in MMIsar0) MMI_recnd: assumes A1: $\varphi \longrightarrow A \in \mathbb{R}$
  shows $\varphi \longrightarrow A \in \mathbb{C}$
$\langle proof \rangle$

lemma (in MMIsar0) MMI_elimne0:
  shows if ($A \neq \mathbf{0}$, A , $\mathbf{1}$) $\neq \mathbf{0}$
$\langle proof \rangle$

lemma (in MMIsar0) MMI_addex:
  shows + isASet
$\langle proof \rangle$

lemma (in MMIsar0) MMI_mulex:
  shows $\cdot$ isASet
$\langle proof \rangle$

lemma (in MMIsar0) MMI_adddirt:
  shows ($A \in \mathbb{C} \wedge B \in \mathbb{C} \wedge C \in \mathbb{C}$) $\longrightarrow$
  ((A + B) $\cdot$ C) = ((A $\cdot$ C) + (B $\cdot$ C))
$\langle proof \rangle$

lemma (in MMIsar0) MMI_addcl: assumes A1: $A \in \mathbb{C}$ and
    A2: $B \in \mathbb{C}$
  shows (A + B) $\in \mathbb{C}$
$\langle proof \rangle$

lemma (in MMIsar0) MMI_mulcl: assumes A1: $A \in \mathbb{C}$ and
    A2: $B \in \mathbb{C}$
  shows (A $\cdot$ B) $\in \mathbb{C}$
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙addcom: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$
shows $(A + B) = (B + A)$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙mulcom: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$
shows $(A \cdot B) = (B \cdot A)$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙addass: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$ and
A3: $C \in \mathbb{C}$
shows $((A + B) + C) = (A + (B + C))$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙mulass: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$ and
A3: $C \in \mathbb{C}$
shows $((A \cdot B) \cdot C) = (A \cdot (B \cdot C))$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙adddi: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$ and
A3: $C \in \mathbb{C}$
shows $(A \cdot (B + C)) = ((A \cdot B) + (A \cdot C))$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙adddir: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$ and
A3: $C \in \mathbb{C}$
shows $((A + B) \cdot C) = ((A \cdot C) + (B \cdot C))$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙1cn:
shows $\mathbf{1} \in \mathbb{C}$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙0cn:
shows $\mathbf{0} \in \mathbb{C}$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙addid1: assumes A1: $A \in \mathbb{C}$
shows $(A + \mathbf{0}) = A$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙addid2: assumes A1: $A \in \mathbb{C}$
shows $(\mathbf{0} + A) = A$

$\langle proof \rangle$

lemma (in MMIsar0) MMI_mulid1: assumes A1: $A \in \mathbb{C}$
shows $(A \cdot \mathbf{1}) = A$
$\langle proof \rangle$

lemma (in MMIsar0) MMI_mulid2: assumes A1: $A \in \mathbb{C}$
shows $(\mathbf{1} \cdot A) = A$
$\langle proof \rangle$

lemma (in MMIsar0) MMI_negex: assumes A1: $A \in \mathbb{C}$
shows $\exists\ x \in \mathbb{C}\ .\ (A + x) = \mathbf{0}$
$\langle proof \rangle$

lemma (in MMIsar0) MMI_recex: assumes A1: $A \in \mathbb{C}$ and
A2: $A \neq \mathbf{0}$
shows $\exists\ x \in \mathbb{C}\ .\ (A \cdot x) = \mathbf{1}$
$\langle proof \rangle$

lemma (in MMIsar0) MMI_readdcl: assumes A1: $A \in \mathbb{R}$ and
A2: $B \in \mathbb{R}$
shows $(A + B) \in \mathbb{R}$
$\langle proof \rangle$

lemma (in MMIsar0) MMI_remulcl: assumes A1: $A \in \mathbb{R}$ and
A2: $B \in \mathbb{R}$
shows $(A \cdot B) \in \mathbb{R}$
$\langle proof \rangle$

lemma (in MMIsar0) MMI_addcan: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$ and
A3: $C \in \mathbb{C}$
shows $(A + B) = (A + C) \longleftrightarrow B = C$
$\langle proof \rangle$

lemma (in MMIsar0) MMI_addcan2: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$ and
A3: $C \in \mathbb{C}$
shows $(A + C) = (B + C) \longleftrightarrow A = B$
$\langle proof \rangle$

lemma (in MMIsar0) MMI_addcant:
shows $(A \in \mathbb{C} \wedge B \in \mathbb{C} \wedge C \in \mathbb{C}) \longrightarrow$
$((A + B) = (A + C) \longleftrightarrow B = C)$
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙addcan2t:
shows (A $\in$ $\mathbb{C}$ $\wedge$ B $\in$ $\mathbb{C}$ $\wedge$ C $\in$ $\mathbb{C}$) $\longrightarrow$ ((A + C) = (B + C) $\longleftrightarrow$ A = B)
$\langle$*proof*$\rangle$

lemma (in MMIsar0) MMI˙add12t:
shows (A $\in$ $\mathbb{C}$ $\wedge$ B $\in$ $\mathbb{C}$ $\wedge$ C $\in$ $\mathbb{C}$) $\longrightarrow$ (A + (B + C)) = (B + (A + C))
$\langle$*proof*$\rangle$

lemma (in MMIsar0) MMI˙add23t:
shows (A $\in$ $\mathbb{C}$ $\wedge$ B $\in$ $\mathbb{C}$ $\wedge$ C $\in$ $\mathbb{C}$) $\longrightarrow$ ((A + B) + C) = ((A + C) + B)
$\langle$*proof*$\rangle$

lemma (in MMIsar0) MMI˙add4t:
shows ((A $\in$ $\mathbb{C}$ $\wedge$ B $\in$ $\mathbb{C}$) $\wedge$ (C $\in$ $\mathbb{C}$ $\wedge$ D $\in$ $\mathbb{C}$)) $\longrightarrow$ ((A + B) + (C + D)) = ((A + C) + (B + D))
$\langle$*proof*$\rangle$

lemma (in MMIsar0) MMI˙add42t:
shows ((A $\in$ $\mathbb{C}$ $\wedge$ B $\in$ $\mathbb{C}$) $\wedge$ (C $\in$ $\mathbb{C}$ $\wedge$ D $\in$ $\mathbb{C}$)) $\longrightarrow$ ((A + B) + (C + D)) = ((A + C) + (D + B))
$\langle$*proof*$\rangle$

lemma (in MMIsar0) MMI˙add12: assumes A1: A $\in$ $\mathbb{C}$ and
A2: B $\in$ $\mathbb{C}$ and
A3: C $\in$ $\mathbb{C}$
shows (A + (B + C)) = (B + (A + C))
$\langle$*proof*$\rangle$

lemma (in MMIsar0) MMI˙add23: assumes A1: A $\in$ $\mathbb{C}$ and
A2: B $\in$ $\mathbb{C}$ and
A3: C $\in$ $\mathbb{C}$
shows ((A + B) + C) = ((A + C) + B)
$\langle$*proof*$\rangle$

lemma (in MMIsar0) MMI˙add4: assumes A1: A $\in$ $\mathbb{C}$ and
A2: B $\in$ $\mathbb{C}$ and
A3: C $\in$ $\mathbb{C}$ and
A4: D $\in$ $\mathbb{C}$
shows ((A + B) + (C + D)) = ((A + C) + (B + D))
$\langle$*proof*$\rangle$

lemma (in MMIsar0) MMI˙add42: assumes A1: A $\in$ $\mathbb{C}$ and
A2: B $\in$ $\mathbb{C}$ and

A3: C $\in \mathbb{C}$ and
A4: D $\in \mathbb{C}$
shows ((A + B) + (C + D)) =
((A + C) + (D + B))
⟨*proof*⟩

lemma (in MMIsar0) MMI˙addid2t:
shows A $\in \mathbb{C} \longrightarrow$ (**0** + A) = A
⟨*proof*⟩

lemma (in MMIsar0) MMI˙peano2cn:
shows A $\in \mathbb{C} \longrightarrow$ (A + **1**) $\in \mathbb{C}$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙peano2re:
shows A $\in \mathbb{R} \longrightarrow$ (A + **1**) $\in \mathbb{R}$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙negeu: assumes A1: A $\in \mathbb{C}$ and
A2: B $\in \mathbb{C}$
shows $\exists!$ x . x $\in \mathbb{C} \wedge$ (A + x) = B
⟨*proof*⟩

lemma (in MMIsar0) MMI˙subval: assumes A $\in \mathbb{C}$ B $\in \mathbb{C}$
shows A - B = $\bigcup$ – x $\in \mathbb{C}$. B + x = A ˝
⟨*proof*⟩

lemma (in MMIsar0) MMI˙df˙neg: shows (- A) = **0** - A
⟨*proof*⟩

lemma (in MMIsar0) MMI˙negeq:
shows A = B $\longrightarrow$ (-A) = (- B)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙negeqi: assumes A1: A = B
shows (- A) = (-B)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙negeqd: assumes A1: $\varphi \longrightarrow$ A = B
shows $\varphi \longrightarrow$ (-A) = (-B)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙hbneg: assumes A1: y $\in$ A $\longrightarrow$ ($\forall$ x . y $\in$ A)
shows y $\in$ ((- A)) $\longrightarrow$ ($\forall$ x . (y $\in$ ((- A))))
⟨*proof*⟩

lemma (in MMIsar0) MMI˙minusex:
shows ((- A)) isASet ⟨*proof*⟩

lemma (in MMIsar0) MMI˙subcl: assumes A1: A $\in \mathbb{C}$ and
A2: B $\in \mathbb{C}$
shows (A - B) $\in \mathbb{C}$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙subclt:
shows (A $\in \mathbb{C}$ $\wedge$ B $\in \mathbb{C}$) $\longrightarrow$ (A - B) $\in \mathbb{C}$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙negclt:
shows A $\in \mathbb{C}$ $\longrightarrow$ ((- A)) $\in \mathbb{C}$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙negcl: assumes A1: A $\in \mathbb{C}$
shows ((- A)) $\in \mathbb{C}$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙subadd: assumes A1: A $\in \mathbb{C}$ and
A2: B $\in \mathbb{C}$ and
A3: C $\in \mathbb{C}$
shows (A - B) = C $\longleftrightarrow$ (B + C) = A
⟨*proof*⟩

lemma (in MMIsar0) MMI˙subsub23: assumes A1: A $\in \mathbb{C}$ and
A2: B $\in \mathbb{C}$ and
A3: C $\in \mathbb{C}$
shows (A - B) = C $\longleftrightarrow$ (A - C) = B
⟨*proof*⟩

lemma (in MMIsar0) MMI˙subaddt:
shows (A $\in \mathbb{C}$ $\wedge$ B $\in \mathbb{C}$ $\wedge$ C $\in \mathbb{C}$) $\longrightarrow$ ((A - B) = C $\longleftrightarrow$
(B + C) = A)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙pncan3t:
shows (A ∈ ℂ ∧ B ∈ ℂ) ⟶ (A + (B - A)) = B
⟨*proof*⟩

lemma (in MMIsar0) MMI˙pncan3: assumes A1: A ∈ ℂ and
A2: B ∈ ℂ
shows (A + (B - A)) = B
⟨*proof*⟩

lemma (in MMIsar0) MMI˙negidt:
shows A ∈ ℂ ⟶ (A + ((- A))) = **0**
⟨*proof*⟩

lemma (in MMIsar0) MMI˙negid: assumes A1: A ∈ ℂ
shows (A + ((- A))) = **0**
⟨*proof*⟩

lemma (in MMIsar0) MMI˙negsub: assumes A1: A ∈ ℂ and
A2: B ∈ ℂ
shows (A + ((- B))) = (A - B)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙negsubt:
shows (A ∈ ℂ ∧ B ∈ ℂ) ⟶ (A + ((- B))) = (A - B)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙addsubasst:
shows (A ∈ ℂ ∧ B ∈ ℂ ∧ C ∈ ℂ) ⟶ ((A + B) - C) =
(A + (B - C))
⟨*proof*⟩

lemma (in MMIsar0) MMI˙addsubt:
shows (A ∈ ℂ ∧ B ∈ ℂ ∧ C ∈ ℂ) ⟶ ((A + B) - C) =
((A - C) + B)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙addsub12t:
shows (A ∈ ℂ ∧ B ∈ ℂ ∧ C ∈ ℂ) ⟶ (A + (B - C)) =
(B + (A - C))
⟨*proof*⟩

lemma (in MMIsar0) MMI˙addsubass: assumes A1: A ∈ ℂ and
A2: B ∈ ℂ and
A3: C ∈ ℂ
shows ((A + B) - C) = (A + (B - C))
⟨*proof*⟩

lemma (in MMIsar0) MMI˙addsub: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$ and
A3: $C \in \mathbb{C}$
shows $((A + B) - C) = ((A - C) + B)$
⟨proof⟩

lemma (in MMIsar0) MMI˙2addsubt:
shows $((A \in \mathbb{C} \wedge B \in \mathbb{C}) \wedge (C \in \mathbb{C} \wedge D \in \mathbb{C})) \longrightarrow$
$(((A + B) + C) - D) = (((A + C) - D) + B)$
⟨proof⟩

lemma (in MMIsar0) MMI˙negneg: assumes A1: $A \in \mathbb{C}$
shows $(- ((- A))) = A$
⟨proof⟩

lemma (in MMIsar0) MMI˙subid: assumes A1: $A \in \mathbb{C}$
shows $(A - A) = \mathbf{0}$
⟨proof⟩

lemma (in MMIsar0) MMI˙subid1: assumes A1: $A \in \mathbb{C}$
shows $(A - \mathbf{0}) = A$
⟨proof⟩

lemma (in MMIsar0) MMI˙negnegt:
shows $A \in \mathbb{C} \longrightarrow (- ((- A))) = A$
⟨proof⟩

lemma (in MMIsar0) MMI˙subnegt:
shows $(A \in \mathbb{C} \wedge B \in \mathbb{C}) \longrightarrow (A - ((- B))) = (A + B)$
⟨proof⟩

lemma (in MMIsar0) MMI˙subidt:
shows $A \in \mathbb{C} \longrightarrow (A - A) = \mathbf{0}$
⟨proof⟩

lemma (in MMIsar0) MMI˙subid1t:
shows $A \in \mathbb{C} \longrightarrow (A - \mathbf{0}) = A$
⟨proof⟩

lemma (in MMIsar0) MMI˙pncant:
shows $(A \in \mathbb{C} \wedge B \in \mathbb{C}) \longrightarrow ((A + B) - B) = A$
⟨proof⟩

lemma (in MMIsar0) MMI˙pncan2t:
shows $(A \in \mathbb{C} \wedge B \in \mathbb{C}) \longrightarrow ((A + B) - A) = B$
⟨proof⟩

lemma (in MMIsar0) MMI˙npcant:

shows (A ∈ ℂ ∧ B ∈ ℂ) ⟶ ((A - B) + B) = A
⟨*proof*⟩

lemma (in MMIsar0) MMI˙npncant:
shows (A ∈ ℂ ∧ B ∈ ℂ ∧ C ∈ ℂ) ⟶
((A - B) + (B - C)) = (A - C)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙nppcant:
shows (A ∈ ℂ ∧ B ∈ ℂ ∧ C ∈ ℂ) ⟶
(((A - B) + C) + B) = (A + C)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙subneg: assumes A1: A ∈ ℂ and
A2: B ∈ ℂ
shows (A - ((- B))) = (A + B)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙subeq0: assumes A1: A ∈ ℂ and
A2: B ∈ ℂ
shows (A - B) = **0** ⟷ A = B
⟨*proof*⟩

lemma (in MMIsar0) MMI˙neg11: assumes A1: A ∈ ℂ and
A2: B ∈ ℂ
shows ((- A)) = ((- B)) ⟷ A = B
⟨*proof*⟩

lemma (in MMIsar0) MMI˙negcon1: assumes A1: A ∈ ℂ and
A2: B ∈ ℂ
shows ((- A)) = B ⟷ ((- B)) = A
⟨*proof*⟩

lemma (in MMIsar0) MMI˙negcon2: assumes A1: A ∈ ℂ and
A2: B ∈ ℂ
shows A = ((- B)) ⟷ B = ((- A))
⟨*proof*⟩

lemma (in MMIsar0) MMI˙neg11t:
shows (A ∈ ℂ ∧ B ∈ ℂ) ⟶ (((- A)) = ((- B)) ⟷ A = B)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙negcon1t:
shows (A ∈ ℂ ∧ B ∈ ℂ) ⟶ (((- A)) = B ⟷ ((- B)) = A)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙negcon2t:

shows (A ∈ ℂ ∧ B ∈ ℂ) ⟶ (A = ((- B)) ⟷ B = ((- A)))
⟨*proof*⟩

lemma (in MMIsar0) MMI˙subcant:
shows (A ∈ ℂ ∧ B ∈ ℂ ∧ C ∈ ℂ) ⟶ ((A - B) =
(A - C) ⟷ B = C)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙subcan2t:
shows (A ∈ ℂ ∧ B ∈ ℂ ∧ C ∈ ℂ) ⟶
((A - C) = (B - C) ⟷ A = B)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙subcan: assumes A1: A ∈ ℂ and
A2: B ∈ ℂ and
A3: C ∈ ℂ
shows (A - B) = (A - C) ⟷ B = C
⟨*proof*⟩

lemma (in MMIsar0) MMI˙subcan2: assumes A1: A ∈ ℂ and
A2: B ∈ ℂ and
A3: C ∈ ℂ
shows (A - C) = (B - C) ⟷ A = B
⟨*proof*⟩

lemma (in MMIsar0) MMI˙subeq0t:
shows (A ∈ ℂ ∧ B ∈ ℂ) ⟶ ((A - B) = **0** ⟷ A = B)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙neg0:
shows (- **0**) = **0**
⟨*proof*⟩

lemma (in MMIsar0) MMI˙renegcl: assumes A1: A ∈ ℝ
shows ((- A)) ∈ ℝ
⟨*proof*⟩

lemma (in MMIsar0) MMI˙renegclt:
shows A ∈ ℝ ⟶ ((- A)) ∈ ℝ
⟨*proof*⟩

lemma (in MMIsar0) MMI˙resubclt:
shows (A ∈ ℝ ∧ B ∈ ℝ) ⟶ (A - B) ∈ ℝ
⟨*proof*⟩

lemma (in MMIsar0) MMI˙resubcl: assumes A1: A ∈ ℝ and
A2: B ∈ ℝ

shows (A - B) $\in$ $\mathbb{R}$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙0re:
shows $\mathbf{0} \in \mathbb{R}$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙mulid2t:
shows A $\in$ $\mathbb{C}$ $\longrightarrow$ ($\mathbf{1}$ $\cdot$ A) = A
⟨*proof*⟩

lemma (in MMIsar0) MMI˙mul12t:
shows (A $\in$ $\mathbb{C}$ $\wedge$ B $\in$ $\mathbb{C}$ $\wedge$ C $\in$ $\mathbb{C}$) $\longrightarrow$ (A $\cdot$ (B $\cdot$ C)) =
(B $\cdot$ (A $\cdot$ C))
⟨*proof*⟩

lemma (in MMIsar0) MMI˙mul23t:
shows (A $\in$ $\mathbb{C}$ $\wedge$ B $\in$ $\mathbb{C}$ $\wedge$ C $\in$ $\mathbb{C}$) $\longrightarrow$ ((A $\cdot$ B) $\cdot$ C) =
((A $\cdot$ C) $\cdot$ B)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙mul4t:
shows ((A $\in$ $\mathbb{C}$ $\wedge$ B $\in$ $\mathbb{C}$) $\wedge$ (C $\in$ $\mathbb{C}$ $\wedge$ D $\in$ $\mathbb{C}$)) $\longrightarrow$
((A $\cdot$ B) $\cdot$ (C $\cdot$ D)) = ((A $\cdot$ C) $\cdot$ (B $\cdot$ D))
⟨*proof*⟩

lemma (in MMIsar0) MMI˙muladdt:
shows ((A $\in$ $\mathbb{C}$ $\wedge$ B $\in$ $\mathbb{C}$) $\wedge$ (C $\in$ $\mathbb{C}$ $\wedge$ D $\in$ $\mathbb{C}$)) $\longrightarrow$
((A + B) $\cdot$ (C + D)) =
(((A $\cdot$ C) + (D $\cdot$ B)) + ((A $\cdot$ D) + (C $\cdot$ B)))
⟨*proof*⟩

lemma (in MMIsar0) MMI˙muladd11t:
shows (A $\in$ $\mathbb{C}$ $\wedge$ B $\in$ $\mathbb{C}$) $\longrightarrow$ (($\mathbf{1}$ + A) $\cdot$ ($\mathbf{1}$ + B)) =
(($\mathbf{1}$ + A) + (B + (A $\cdot$ B)))
⟨*proof*⟩

lemma (in MMIsar0) MMI˙mul12: assumes A1: A $\in$ $\mathbb{C}$ and
A2: B $\in$ $\mathbb{C}$ and
A3: C $\in$ $\mathbb{C}$
shows (A $\cdot$ (B $\cdot$ C)) = (B $\cdot$ (A $\cdot$ C))
⟨*proof*⟩

lemma (in MMIsar0) MMI˙mul23: assumes A1: A $\in$ $\mathbb{C}$ and
A2: B $\in$ $\mathbb{C}$ and

A3: C ∈ ℂ
shows ((A · B) · C) = ((A · C) · B)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙mul4: assumes A1: A ∈ ℂ and
A2: B ∈ ℂ and
A3: C ∈ ℂ and
A4: D ∈ ℂ
shows ((A · B) · (C · D)) = ((A · C) · (B · D))
⟨*proof*⟩

lemma (in MMIsar0) MMI˙muladd: assumes A1: A ∈ ℂ and
A2: B ∈ ℂ and
A3: C ∈ ℂ and
A4: D ∈ ℂ
shows ((A + B) · (C + D)) =
(((A · C) + (D · B)) + ((A · D) + (C · B)))
⟨*proof*⟩

lemma (in MMIsar0) MMI˙subdit:
shows (A ∈ ℂ ∧ B ∈ ℂ ∧ C ∈ ℂ) ⟶
(A · (B - C)) = ((A · B) - (A · C))
⟨*proof*⟩

lemma (in MMIsar0) MMI˙subdirt:
shows (A ∈ ℂ ∧ B ∈ ℂ ∧ C ∈ ℂ) ⟶
((A - B) · C) = ((A · C) - (B · C))
⟨*proof*⟩

lemma (in MMIsar0) MMI˙subdi: assumes A1: A ∈ ℂ and
A2: B ∈ ℂ and
A3: C ∈ ℂ
shows (A · (B - C)) = ((A · B) - (A · C))
⟨*proof*⟩

lemma (in MMIsar0) MMI˙subdir: assumes A1: A ∈ ℂ and
A2: B ∈ ℂ and
A3: C ∈ ℂ
shows ((A - B) · C) = ((A · C) - (B · C))
⟨*proof*⟩

lemma (in MMIsar0) MMI˙mul01: assumes A1: A ∈ ℂ
shows (A · **0**) = **0**
⟨*proof*⟩

lemma (in MMIsar0) MMI˙mul02: assumes A1: A ∈ ℂ
shows (**0** · A) = **0**
⟨*proof*⟩

lemma (in MMIsar0) MMI˙1p1times: assumes A1: $A \in \mathbb{C}$
shows (($\mathbf{1} + \mathbf{1}$) $\cdot$ A) = (A + A)
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙mul01t:
shows $A \in \mathbb{C} \longrightarrow$ (A $\cdot$ $\mathbf{0}$) = $\mathbf{0}$
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙mul02t:
shows $A \in \mathbb{C} \longrightarrow$ ($\mathbf{0}$ $\cdot$ A) = $\mathbf{0}$
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙mulneg1: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$
shows (((- A)) $\cdot$ B) = (- (A $\cdot$ B))
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙mulneg2: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$
shows (A $\cdot$ ((- B))) =
(- (A $\cdot$ B))
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙mul2neg: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$
shows (((- A)) $\cdot$ ((- B))) =
(A $\cdot$ B)
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙negdi: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$
shows (- (A + B)) =
(((- A)) + ((- B)))
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙negsubdi: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$
shows (- (A - B)) =
(((- A)) + B)
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙negsubdi2: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$
shows (- (A - B)) = (B - A)
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙mulneg1t:
shows (A $\in$ $\mathbb{C}$ $\wedge$ B $\in$ $\mathbb{C}$) $\longrightarrow$
(((- A)) $\cdot$ B) =
(- (A $\cdot$ B))
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙mulneg2t:
shows (A $\in$ $\mathbb{C}$ $\wedge$ B $\in$ $\mathbb{C}$) $\longrightarrow$
(A $\cdot$ ((- B))) =
(- (A $\cdot$ B))
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙mulneg12t:
shows (A $\in$ $\mathbb{C}$ $\wedge$ B $\in$ $\mathbb{C}$) $\longrightarrow$
(((- A)) $\cdot$ B) =
(A $\cdot$ ((- B)))
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙mul2negt:
shows (A $\in$ $\mathbb{C}$ $\wedge$ B $\in$ $\mathbb{C}$) $\longrightarrow$
(((- A)) $\cdot$ ((- B))) =
(A $\cdot$ B)
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙negdit:
shows (A $\in$ $\mathbb{C}$ $\wedge$ B $\in$ $\mathbb{C}$) $\longrightarrow$
(- (A + B)) =
(((- A)) + ((- B)))
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙negdi2t:
shows (A $\in$ $\mathbb{C}$ $\wedge$ B $\in$ $\mathbb{C}$) $\longrightarrow$
(- (A + B)) = (((- A)) - B)
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙negsubdit:
shows (A $\in$ $\mathbb{C}$ $\wedge$ B $\in$ $\mathbb{C}$) $\longrightarrow$
(- (A - B)) = (((- A)) + B)
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙negsubdi2t:
shows (A $\in$ $\mathbb{C}$ $\wedge$ B $\in$ $\mathbb{C}$) $\longrightarrow$
(- (A - B)) = (B - A)
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙subsub2t:
shows (A $\in$ $\mathbb{C}$ $\wedge$ B $\in$ $\mathbb{C}$ $\wedge$ C $\in$ $\mathbb{C}$) $\longrightarrow$

(A - (B - C)) = (A + (C - B))
⟨*proof*⟩

lemma (in MMIsar0) MMI˙subsubt:
  shows (A ∈ ℂ ∧ B ∈ ℂ ∧ C ∈ ℂ) ⟶
(A - (B - C)) = ((A - B) + C)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙subsub3t:
  shows (A ∈ ℂ ∧ B ∈ ℂ ∧ C ∈ ℂ) ⟶
(A - (B - C)) = ((A + C) - B)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙subsub4t:
  shows (A ∈ ℂ ∧ B ∈ ℂ ∧ C ∈ ℂ) ⟶
((A - B) - C) = (A - (B + C))
⟨*proof*⟩

lemma (in MMIsar0) MMI˙sub23t:
  shows (A ∈ ℂ ∧ B ∈ ℂ ∧ C ∈ ℂ) ⟶
((A - B) - C) = ((A - C) - B)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙nnncant:
  shows (A ∈ ℂ ∧ B ∈ ℂ ∧ C ∈ ℂ) ⟶
((A - (B - C)) - C) = (A - B)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙nnncan1t:
  shows (A ∈ ℂ ∧ B ∈ ℂ ∧ C ∈ ℂ) ⟶
((A - B) - (A - C)) = (C - B)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙nnncan2t:
  shows (A ∈ ℂ ∧ B ∈ ℂ ∧ C ∈ ℂ) ⟶
((A - C) - (B - C)) = (A - B)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙nncant:
  shows (A ∈ ℂ ∧ B ∈ ℂ) ⟶
(A - (A - B)) = B
⟨*proof*⟩

lemma (in MMIsar0) MMI˙nppcan2t:
  shows (A ∈ ℂ ∧ B ∈ ℂ ∧ C ∈ ℂ) ⟶
((A - (B + C)) + C) = (A - B)
⟨*proof*⟩

lemma (in MMIsar0) MMI_mulm1t:
  shows $A \in \mathbb{C} \longrightarrow ((- \mathbf{1}) \cdot A) = ((- A))$
⟨*proof*⟩

lemma (in MMIsar0) MMI_mulm1: assumes A1: $A \in \mathbb{C}$
  shows $((- \mathbf{1}) \cdot A) = ((- A))$
⟨*proof*⟩

lemma (in MMIsar0) MMI_sub4t:
  shows $((A \in \mathbb{C} \land B \in \mathbb{C}) \land (C \in \mathbb{C} \land D \in \mathbb{C})) \longrightarrow$
$((A + B) - (C + D)) =$
$((A - C) + (B - D))$
⟨*proof*⟩

lemma (in MMIsar0) MMI_sub4: assumes A1: $A \in \mathbb{C}$ and
  A2: $B \in \mathbb{C}$ and
  A3: $C \in \mathbb{C}$ and
  A4: $D \in \mathbb{C}$
  shows $((A + B) - (C + D)) =$
$((A - C) + (B - D))$
⟨*proof*⟩

lemma (in MMIsar0) MMI_mulsubt:
  shows $((A \in \mathbb{C} \land B \in \mathbb{C}) \land (C \in \mathbb{C} \land D \in \mathbb{C})) \longrightarrow$
$((A - B) \cdot (C - D)) =$
$(((A \cdot C) + (D \cdot B)) - ((A \cdot D) + (C \cdot B)))$
⟨*proof*⟩

lemma (in MMIsar0) MMI_pnpcant:
  shows $(A \in \mathbb{C} \land B \in \mathbb{C} \land C \in \mathbb{C}) \longrightarrow$
$((A + B) - (A + C)) = (B - C)$
⟨*proof*⟩

lemma (in MMIsar0) MMI_pnpcan2t:
  shows $(A \in \mathbb{C} \land B \in \mathbb{C} \land C \in \mathbb{C}) \longrightarrow$
$((A + C) - (B + C)) = (A - B)$
⟨*proof*⟩

lemma (in MMIsar0) MMI_pnncant:
  shows $(A \in \mathbb{C} \land B \in \mathbb{C} \land C \in \mathbb{C}) \longrightarrow$
$((A + B) - (A - C)) = (B + C)$
⟨*proof*⟩

lemma (in MMIsar0) MMI_ppncant:
  shows $(A \in \mathbb{C} \land B \in \mathbb{C} \land C \in \mathbb{C}) \longrightarrow$
$((A + B) + (C - B)) = (A + C)$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙pnncan: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$ and
A3: $C \in \mathbb{C}$
shows $((A + B) - (A - C)) = (B + C)$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙mulcan: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$ and
A3: $C \in \mathbb{C}$ and
A4: $A \neq \mathbf{0}$
shows $(A \cdot B) = (A \cdot C) \longleftrightarrow B = C$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙mulcant2: assumes A1: $A \neq \mathbf{0}$
shows $(A \in \mathbb{C} \wedge B \in \mathbb{C} \wedge C \in \mathbb{C}) \longrightarrow$
$((A \cdot B) = (A \cdot C) \longleftrightarrow B = C)$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙mulcant:
shows $((A \in \mathbb{C} \wedge B \in \mathbb{C} \wedge C \in \mathbb{C}) \wedge A \neq \mathbf{0}) \longrightarrow$
$((A \cdot B) = (A \cdot C) \longleftrightarrow B = C)$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙mulcan2t:
shows $((A \in \mathbb{C} \wedge B \in \mathbb{C} \wedge C \in \mathbb{C}) \wedge C \neq \mathbf{0}) \longrightarrow$
$((A \cdot C) = (B \cdot C) \longleftrightarrow A = B)$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙mul0or: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$
shows $(A \cdot B) = \mathbf{0} \longleftrightarrow (A = \mathbf{0} \vee B = \mathbf{0})$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙msq0: assumes A1: $A \in \mathbb{C}$
shows $(A \cdot A) = \mathbf{0} \longleftrightarrow A = \mathbf{0}$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙mul0ort:
shows $(A \in \mathbb{C} \wedge B \in \mathbb{C}) \longrightarrow$
$((A \cdot B) = \mathbf{0} \longleftrightarrow (A = \mathbf{0} \vee B = \mathbf{0}))$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙muln0bt:
shows $(A \in \mathbb{C} \wedge B \in \mathbb{C}) \longrightarrow$
$((A \neq \mathbf{0} \wedge B \neq \mathbf{0}) \longleftrightarrow (A \cdot B) \neq \mathbf{0})$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙muln0: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$ and
A3: $A \neq \mathbf{0}$ and
A4: $B \neq \mathbf{0}$
shows $(A \cdot B) \neq \mathbf{0}$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙receu: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$ and
A3: $A \neq \mathbf{0}$
shows $\exists! \ x . x \in \mathbb{C} \wedge (A \cdot x) = B$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divval: assumes $A \in \mathbb{C}$ $B \in \mathbb{C}$ $B \neq \mathbf{0}$
shows $A / B = \bigcup - x \in \mathbb{C} . B \cdot x = A$ ″
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divmul: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$ and
A3: $C \in \mathbb{C}$ and
A4: $B \neq \mathbf{0}$
shows $(A / B) = C \longleftrightarrow (B \cdot C) = A$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divmulz: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$ and
A3: $C \in \mathbb{C}$
shows $B \neq \mathbf{0} \longrightarrow$
$((A / B) = C \longleftrightarrow (B \cdot C) = A)$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divmult:
shows $((A \in \mathbb{C} \wedge B \in \mathbb{C} \wedge C \in \mathbb{C}) \wedge B \neq \mathbf{0}) \longrightarrow$
$((A / B) = C \longleftrightarrow (B \cdot C) = A)$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divmul2t:
shows $((A \in \mathbb{C} \wedge B \in \mathbb{C} \wedge C \in \mathbb{C}) \wedge B \neq \mathbf{0}) \longrightarrow$
$((A / B) = C \longleftrightarrow A = (B \cdot C))$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divmul3t:
shows $((A \in \mathbb{C} \wedge B \in \mathbb{C} \wedge C \in \mathbb{C}) \wedge B \neq \mathbf{0}) \longrightarrow$

$((A / B) = C \longleftrightarrow A = (C \cdot B))$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divcl: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$ and
A3: $B \neq \mathbf{0}$
shows $(A / B) \in \mathbb{C}$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divclz: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$
shows $B \neq \mathbf{0} \longrightarrow (A / B) \in \mathbb{C}$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divclt:
shows $(A \in \mathbb{C} \wedge B \in \mathbb{C} \wedge B \neq \mathbf{0}) \longrightarrow$
$(A / B) \in \mathbb{C}$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙reccl: assumes A1: $A \in \mathbb{C}$ and
A2: $A \neq \mathbf{0}$
shows $(\mathbf{1} / A) \in \mathbb{C}$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙recclz: assumes A1: $A \in \mathbb{C}$
shows $A \neq \mathbf{0} \longrightarrow (\mathbf{1} / A) \in \mathbb{C}$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙recclt:
shows $(A \in \mathbb{C} \wedge A \neq \mathbf{0}) \longrightarrow (\mathbf{1} / A) \in \mathbb{C}$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divcan2: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$ and
A3: $A \neq \mathbf{0}$
shows $(A \cdot (B / A)) = B$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divcan1: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$ and
A3: $A \neq \mathbf{0}$
shows $((B / A) \cdot A) = B$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divcan1z: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$

shows A $\neq$ **0** $\longrightarrow$ ((B / A) $\cdot$ A) = B
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divcan2z: assumes A1: A $\in \mathbb{C}$ and
A2: B $\in \mathbb{C}$
shows A $\neq$ **0** $\longrightarrow$ (A $\cdot$ (B / A)) = B
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divcan1t:
shows (A $\in \mathbb{C}$ $\wedge$ B $\in \mathbb{C}$ $\wedge$ A $\neq$ **0**) $\longrightarrow$
((B / A) $\cdot$ A) = B
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divcan2t:
shows (A $\in \mathbb{C}$ $\wedge$ B $\in \mathbb{C}$ $\wedge$ A $\neq$ **0**) $\longrightarrow$
(A $\cdot$ (B / A)) = B
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divne0bt:
shows (A $\in \mathbb{C}$ $\wedge$ B $\in \mathbb{C}$ $\wedge$ B $\neq$ **0**) $\longrightarrow$
(A $\neq$ **0** $\longleftrightarrow$ (A / B) $\neq$ **0**)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divne0: assumes A1: A $\in \mathbb{C}$ and
A2: B $\in \mathbb{C}$ and
A3: A $\neq$ **0** and
A4: B $\neq$ **0**
shows (A / B) $\neq$ **0**
⟨*proof*⟩

lemma (in MMIsar0) MMI˙recne0z: assumes A1: A $\in \mathbb{C}$
shows A $\neq$ **0** $\longrightarrow$ (**1** / A) $\neq$ **0**
⟨*proof*⟩

lemma (in MMIsar0) MMI˙recne0t:
shows (A $\in \mathbb{C}$ $\wedge$ A $\neq$ **0**) $\longrightarrow$ (**1** / A) $\neq$ **0**
⟨*proof*⟩

lemma (in MMIsar0) MMI˙recid: assumes A1: A $\in \mathbb{C}$ and
A2: A $\neq$ **0**
shows (A $\cdot$ (**1** / A)) = **1**
⟨*proof*⟩

lemma (in MMIsar0) MMI˙recidz: assumes A1: A $\in \mathbb{C}$
shows A $\neq$ **0** $\longrightarrow$ (A $\cdot$ (**1** / A)) = **1**
⟨*proof*⟩

lemma (in MMIsar0) MMI˙recidt:
shows (A $\in \mathbb{C} \wedge$ A $\neq \mathbf{0}$) $\longrightarrow$
(A $\cdot$ ($\mathbf{1}$ / A)) = $\mathbf{1}$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙recid2t:
shows (A $\in \mathbb{C} \wedge$ A $\neq \mathbf{0}$) $\longrightarrow$
(($\mathbf{1}$ / A) $\cdot$ A) = $\mathbf{1}$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divrec: assumes A1: A $\in \mathbb{C}$ and
A2: B $\in \mathbb{C}$ and
A3: B $\neq \mathbf{0}$
shows (A / B) = (A $\cdot$ ($\mathbf{1}$ / B))
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divrecz: assumes A1: A $\in \mathbb{C}$ and
A2: B $\in \mathbb{C}$
shows B $\neq \mathbf{0} \longrightarrow$ (A / B) = (A $\cdot$ ($\mathbf{1}$ / B))
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divrect:
shows (A $\in \mathbb{C} \wedge$ B $\in \mathbb{C} \wedge$ B $\neq \mathbf{0}$) $\longrightarrow$
(A / B) = (A $\cdot$ ($\mathbf{1}$ / B))
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divrec2t:
shows (A $\in \mathbb{C} \wedge$ B $\in \mathbb{C} \wedge$ B $\neq \mathbf{0}$) $\longrightarrow$
(A / B) = (($\mathbf{1}$ / B) $\cdot$ A)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divasst:
shows ((A $\in \mathbb{C} \wedge$ B $\in \mathbb{C} \wedge$ C $\in \mathbb{C}$) $\wedge$ C $\neq \mathbf{0}$) $\longrightarrow$
((A $\cdot$ B) / C) = (A $\cdot$ (B / C))
⟨*proof*⟩

lemma (in MMIsar0) MMI˙div23t:
shows ((A $\in \mathbb{C} \wedge$ B $\in \mathbb{C} \wedge$ C $\in \mathbb{C}$) $\wedge$ C $\neq \mathbf{0}$) $\longrightarrow$
((A $\cdot$ B) / C) = ((A / C) $\cdot$ B)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙div13t:
shows ((A $\in \mathbb{C} \wedge$ B $\in \mathbb{C} \wedge$ C $\in \mathbb{C}$) $\wedge$ B $\neq \mathbf{0}$) $\longrightarrow$
((A / B) $\cdot$ C) = ((C / B) $\cdot$ A)
⟨*proof*⟩

lemma (in MMIsar0) MMI˙div12t:

shows ((A $\in \mathbb{C}$ $\wedge$ B $\in \mathbb{C}$ $\wedge$ C $\in \mathbb{C}$) $\wedge$ C $\neq \mathbf{0}$) $\longrightarrow$
(A $\cdot$ (B / C)) = (B $\cdot$ (A / C))
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙divassz: assumes A1: A $\in \mathbb{C}$ and
A2: B $\in \mathbb{C}$ and
A3: C $\in \mathbb{C}$
shows C $\neq \mathbf{0}$ $\longrightarrow$
((A $\cdot$ B) / C) = (A $\cdot$ (B / C))
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙divass: assumes A1: A $\in \mathbb{C}$ and
A2: B $\in \mathbb{C}$ and
A3: C $\in \mathbb{C}$ and
A4: C $\neq \mathbf{0}$
shows ((A $\cdot$ B) / C) = (A $\cdot$ (B / C))
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙divdir: assumes A1: A $\in \mathbb{C}$ and
A2: B $\in \mathbb{C}$ and
A3: C $\in \mathbb{C}$ and
A4: C $\neq \mathbf{0}$
shows ((A + B) / C) =
((A / C) + (B / C))
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙div23: assumes A1: A $\in \mathbb{C}$ and
A2: B $\in \mathbb{C}$ and
A3: C $\in \mathbb{C}$ and
A4: C $\neq \mathbf{0}$
shows ((A $\cdot$ B) / C) = ((A / C) $\cdot$ B)
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙divdirz: assumes A1: A $\in \mathbb{C}$ and
A2: B $\in \mathbb{C}$ and
A3: C $\in \mathbb{C}$
shows C $\neq \mathbf{0}$ $\longrightarrow$
((A + B) / C) =
((A / C) + (B / C))
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙divdirt:
shows ((A $\in \mathbb{C}$ $\wedge$ B $\in \mathbb{C}$ $\wedge$ C $\in \mathbb{C}$) $\wedge$ C $\neq \mathbf{0}$) $\longrightarrow$
((A + B) / C) =
((A / C) + (B / C))
$\langle proof \rangle$

lemma (in MMIsar0) MMI˙divcan3: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$ and
A3: $A \neq \mathbf{0}$
shows $((A \cdot B) / A) = B$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divcan4: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$ and
A3: $A \neq \mathbf{0}$
shows $((B \cdot A) / A) = B$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divcan3z: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$
shows $A \neq \mathbf{0} \longrightarrow ((A \cdot B) / A) = B$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divcan4z: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$
shows $A \neq \mathbf{0} \longrightarrow ((B \cdot A) / A) = B$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divcan3t:
shows $(A \in \mathbb{C} \wedge B \in \mathbb{C} \wedge A \neq \mathbf{0}) \longrightarrow$
$((A \cdot B) / A) = B$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divcan4t:
shows $(A \in \mathbb{C} \wedge B \in \mathbb{C} \wedge A \neq \mathbf{0}) \longrightarrow$
$((B \cdot A) / A) = B$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙div11: assumes A1: $A \in \mathbb{C}$ and
A2: $B \in \mathbb{C}$ and
A3: $C \in \mathbb{C}$ and
A4: $C \neq \mathbf{0}$
shows $(A / C) = (B / C) \longleftrightarrow A = B$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙div11t:
shows $(A \in \mathbb{C} \wedge B \in \mathbb{C} \wedge (C \in \mathbb{C} \wedge C \neq \mathbf{0})) \longrightarrow$
$((A / C) = (B / C) \longleftrightarrow A = B)$
⟨*proof*⟩

end

80 Metamath examples

theory MMI˙examples imports MMI˙Complex˙ZF

begin

This theory contains 10 theorems translated from Metamath (with proofs). It is included in the proof document as an illustration of how a translated Metamath proof looks like. The "known_theorems.txt" file included in the IsarMathLib distribution provides a list of all translated facts.

lemma (in MMIsar0) MMI˙dividt:
  shows $(A \in \mathbb{C} \wedge A \neq \mathbf{0}) \longrightarrow (A / A) = \mathbf{1}$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙div0t:
  shows $(A \in \mathbb{C} \wedge A \neq \mathbf{0}) \longrightarrow (\mathbf{0} / A) = \mathbf{0}$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙diveq0t:
  shows $(A \in \mathbb{C} \wedge C \in \mathbb{C} \wedge C \neq \mathbf{0}) \longrightarrow$
 $((A / C) = \mathbf{0} \longleftrightarrow A = \mathbf{0})$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙recrec: assumes A1: $A \in \mathbb{C}$ and
  A2: $A \neq \mathbf{0}$
  shows $(\mathbf{1} / (\mathbf{1} / A)) = A$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divid: assumes A1: $A \in \mathbb{C}$ and
  A2: $A \neq \mathbf{0}$
  shows $(A / A) = \mathbf{1}$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙div0: assumes A1: $A \in \mathbb{C}$ and
  A2: $A \neq \mathbf{0}$
  shows $(\mathbf{0} / A) = \mathbf{0}$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙div1: assumes A1: $A \in \mathbb{C}$
  shows $(A / \mathbf{1}) = A$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙div1t:
  shows $A \in \mathbb{C} \longrightarrow (A / \mathbf{1}) = A$
⟨*proof*⟩

lemma (in MMIsar0) MMI˙divnegt:
  shows $(A \in \mathbb{C} \wedge B \in \mathbb{C} \wedge B \neq \mathbf{0}) \longrightarrow$
 $(- (A / B)) = ((- A) / B)$

⟨*proof*⟩

lemma (in MMIsar0) MMI˙divsubdirt:
shows ((A $\in$ $\mathbb{C}$ $\wedge$ B $\in$ $\mathbb{C}$ $\wedge$ C $\in$ $\mathbb{C}$) $\wedge$ C $\neq$ **0**) $\longrightarrow$
((A - B) / C) =
((A / C) - (B / C))
⟨*proof*⟩

end

81 Metamath interface

theory Metamath˙Interface imports Complex˙ZF MMI˙prelude

begin

This theory contains some lemmas that make it possible to use the theorems translated from Metamath in a the complex0 context.

81.1 MMisar0 and complex0 contexts.

In the section we show a lemma that the assumptions in complex0 context imply the assumptions of the MMIsar0 context. The Metamath˙sampler theory provides examples how this lemma can be used.

The next lemma states that we can use the theorems proven in the MMIsar0 context in the complex0 context. Unfortunately we have to use low level Isabelle methods "rule" and "unfold" in the proof, simp and blast fail on the order axioms.

lemma (in complex0) MMIsar˙valid:
shows MMIsar0($\mathbb{R}$,$\mathbb{C}$,**1**,**0**,i,CplxAdd(R,A),CplxMul(R,A,M),
StrictVersion(CplxROrder(R,A,r)))
⟨*proof*⟩

end

82 Metamath sampler

theory Metamath˙Sampler imports Metamath˙Interface MMI˙Complex˙ZF˙2

begin

The theorems translated from Metamath reside in the MMI˙Complex˙ZF, MMI˙Complex˙ZF˙1 and MMI˙Complex˙ZF˙2 theories. The proofs of these theorems are very verbose and for this reason the theories are not shown in the proof document or the FormaMath.org site. This theory file contains some examples of theorems translated from Metamath and formulated in the complex0 context. This serves two purposes: to give an overview of the material covered in the translated theorems and to provide examples of how to take a translated theorem (proven in the MMIsar0 context) and transfer it to the complex0 context. The typical procedure for moving a theorem from MMIsar0 to complex0 is as follows: First we define certain aliases that map names defined in the complex0 to their corresponding names in the MMIsar0 context. This makes it easy to copy and paste the statement of the theorem as displayed with ProofGeneral. Then we run the Isabelle from ProofGeneral up to the theorem we want to move. When the theorem is verified ProofGeneral displays the statement in the raw set theory notation, stripped from any notation defined in the MMIsar0 locale. This is what we copy to the proof in the complex0 locale. After that we just can write "then have ?thesis by simp" and the simplifier translates the raw set theory notation to the one used in complex0.

82.1 Extended reals and order

In this sectin we import a couple of theorems about the extended real line and the linear order on it.

Metamath uses the set of real numbers extended with $+\infty$ and $-\infty$. The $+\infty$ and $-\infty$ symbols are defined quite arbitrarily as $\mathbb{C}$ and $\{\mathbb{C}\}$, respectively. The next lemma that corresponds to Metamath's renfdisj states that $+\infty$ and $-\infty$ are not elements of $\mathbb{R}$.

lemma (in complex0) renfdisj: shows $\mathbb{R} \cap -+\infty,-\infty'' = 0$
⟨*proof*⟩

The order relation used most often in Metamath is defined on the set of complex reals extended with $+\infty$ and $-\infty$. The next lemma allows to use Metamath's xrltso that states that the ¡ relations is a strict linear order on the extended set.

lemma (in complex0) xrltso: shows ¡ Orders $\mathbb{R}^*$
⟨*proof*⟩

Metamath defines the usual $<$ and $\leq$ ordering relations for the extended real line, including $+\infty$ and $-\infty$.

lemma (in complex0) xrrebndt: assumes A1: $x \in \mathbb{R}^*$
shows $x \in \mathbb{R} \longleftrightarrow (-\infty$ ¡ $x \wedge x$ ¡ $+\infty)$
⟨*proof*⟩

A quite involved inequality.

lemma (in complex0) lt2mul2divt:
assumes A1: $a \in \mathbb{R}$ $b \in \mathbb{R}$ $c \in \mathbb{R}$ $d \in \mathbb{R}$ and

A2: $\mathbf{0}$ ¡ b $\mathbf{0}$ ¡ d
shows a·b ¡ c·d $\longleftrightarrow$ a/d ¡ c/b
⟨*proof*⟩

A real number is smaller than its half iff it is positive.

lemma (in complex0) halfpos: assumes A1: a $\in$ $\mathbb{R}$
shows $\mathbf{0}$ ¡ a $\longleftrightarrow$ a/$\mathbf{2}$ ¡ a
⟨*proof*⟩

One more inequality.

lemma (in complex0) ledivp1t:
assumes A1: a $\in$ $\mathbb{R}$ b $\in$ $\mathbb{R}$ and
A2: $\mathbf{0}$ $\leq$ a $\mathbf{0}$ $\leq$ b
shows (a/(b + $\mathbf{1}$))·b $\leq$ a
⟨*proof*⟩

82.2 Natural real numbers

In standard mathematics natural numbers are treated as a subset of real numbers. From the set theory point of view however those are quite different objects. In this section we talk about "real natural" numbers i.e. the conterpart of natural numbers that is a subset of the reals.

Two ways of saying that there are no natural numbers between n and $n+1$.

lemma (in complex0) no˙nats˙between:
assumes A1: n $\in$ $\mathbb{N}$ k $\in$ $\mathbb{N}$
shows
n$\leq$k $\longleftrightarrow$ n ¡ k+$\mathbf{1}$
n ¡ k $\longleftrightarrow$ n + $\mathbf{1}$ $\leq$ k
⟨*proof*⟩

Metamath has some very complicated and general version of induction on (complex) natural numbers that I can't even understand. As an exercise I derived a more standard version that is imported to the complex0 context below.

lemma (in complex0) cplx˙nat˙ind: assumes A1: ψ($\mathbf{1}$) and
A2: $\forall$ k $\in$ $\mathbb{N}$. ψ(k) $\longrightarrow$ ψ(k+$\mathbf{1}$) and
A3: n $\in$ $\mathbb{N}$
shows ψ(n)
⟨*proof*⟩

Some simple arithmetics.

lemma (in complex0) arith: shows
$\mathbf{2}$ + $\mathbf{2}$ = $\mathbf{4}$
$\mathbf{2}$·$\mathbf{2}$ = $\mathbf{4}$
$\mathbf{3}$·$\mathbf{2}$ = $\mathbf{6}$
$\mathbf{3}$·$\mathbf{3}$ = $\mathbf{9}$
⟨*proof*⟩

82.3 Infimum and supremum in real numbers

Real numbers form a complete ordered field. Here we import a couple of Metamath theorems about supremu and infimum.

If a set S has a smallest element, then the infimum of S belongs to it.

lemma (in complex0) lbinfmcl: assumes A1: S ⊆ ℝ and
  A2: ∃x∈S. ∀y∈S. x ≤ y
  shows Infim(S,ℝ,¡) ∈ S
⟨*proof*⟩

Supremum of any subset of reals that is bounded above is real.

lemma (in complex0) sup˙is˙real:
  assumes A ⊆ ℝ and A ≠ 0 and ∃x∈ℝ. ∀y∈A. y ≤ x
  shows Sup(A,ℝ,¡) ∈ ℝ
⟨*proof*⟩

If a real number is smaller that the supremum of A, then we can find an element of A greater than it.

lemma (in complex0) suprlub:
  assumes A ⊆ℝ and A ≠ 0 and ∃x∈ℝ. ∀y∈A. y ≤ x
  and B ∈ ℝ and B ¡ Sup(A,ℝ,¡)
  shows ∃z∈A. B ¡ z
⟨*proof*⟩

Something a bit more interesting: infimum of a set that is bounded below is real and equal to the minus supremum of the set flipped around zero.

lemma (in complex0) infmsup:
  assumes A ⊆ ℝ and A ≠ 0 and ∃x∈ℝ. ∀y∈A. x ≤ y
  shows
  Infim(A,ℝ,¡) ∈ ℝ
  Infim(A,ℝ,¡) = (-Sup(–z ∈ ℝ. (-z) ∈ A ˝,ℝ,¡))
⟨*proof*⟩

end

References

[1] N. A'Campo. A natural construction for the real numbers. 2003.

[2] R. D. Arthan. The Eudoxus Real Numbers. 2004.

[3] R. Street at al. The Efficient Real Numbers. 2003.

[4] Strecker G.E. Herrlich H. When is ℕ lindelöf? Comment. Math. Univ. Carolinae, 1997.

[5] I. L. Reilly and M. K. Vamanamurthy. Some topological anti-properties. Illinois J. Math., 24:382–389, 1980.