Annals of Formalized Mathematics
Volume **1** (2025), 133–159
DOI: https://doi.org/10.46298/afm.14840

Formalising the local compactness of the adele ring

Salvatore Mercuri

Abstract. The adele ring of a number field is a central object in modern number theory. Its status as a locally compact topological ring is one of the key reasons why. We describe a formal proof that the adele ring of a number field is locally compact implemented in the Lean 4 theorem prover. Our work includes the formalisations of new types, including the completion of a number field at an infinite place, the infinite adele ring and the finite S-adele ring, as well as formal proofs that completions of a number field are locally compact and that their rings of integers at finite places are compact.

MSC 2020: 11R56, 68V20
Keywords: algebraic number theory, adele ring, Lean, mathlib

Contact information:
Salvatore Mercuri: The University of Edinburgh, UK. smercuri@ed.ac.uk

Received: November 25, 2024
Final form: April 29, 2025
Accepted: July 2, 2025

1 Introduction

In number theory, looking locally can give a lot of information about the global picture with which we are primarily concerned. For example, the equation $x^2 + 3y^2 = 2024$ has no rational solution, because any such solution would give a local solution modulo 3 of the equation $x^2 \equiv 2 \pmod 3$, yet 2 is not a square modulo 3. More generally, Hasse's *local-global principle* asserts that, under certain conditions, a global rational solution of an equation exists if and only if there is a real solution and local solutions for every prime p.

The local study of the field $\mathbb{Q}$ of rational numbers at a prime p is systematised in the field $\mathbb{Q}_p$ of p-adic numbers. Just as the real numbers $\mathbb{R}$ are the completion of $\mathbb{Q}$ with respect to the usual absolute value, the field $\mathbb{Q}_p$ is the completion of $\mathbb{Q}$ with respect to the p-adic absolute value. In line with the local-global principle, we might intuitively unify all real and local information into the product $\mathbb{R} \times \prod_p \mathbb{Q}_p$, where p ranges over all primes. The rational adele ring $\mathbb{A}_{\mathbb{Q}}$ is a subring of this product defined by a restriction on the infinite product over p which we detail later. This particular restriction makes the adele ring a *locally compact* topological ring.

In addition to encoding local information, the adele ring's local compactness has led to its widespread use in modern number theory. One particular example is the ability to do Fourier analysis on the adele ring, which Tate [Tat50] used to reformulate Dirichlet L-functions and their functional equations on adeles. These functional equations have normalisation factors which, in the classical setting, appeared artificially but in the adelic setting appear naturally from the real component of the rational adele ring. Tate's thesis is really the theory of automorphic forms on $GL(1)$ and a precursor to the Langlands program where the adele ring analogously plays a foundational role.

Adele rings of general number fields can be defined similarly to the rational adele ring. In this paper, we formalise the proof that the adele ring of a number field is a locally compact topological ring in the Lean theorem prover, `v4.10.0`, with the use of Lean's mathematics library MATHLIB [mat20] at commit `caac5b1`. All hyperlinks to MATHLIB in this paper link to this commit; objects and results introduced after this commit are referenced via their introductory PRs. The reference project code is available on GitHub[1]. To our knowledge, this is the first formalised proof of the local compactness of the adele ring in any interactive theorem prover, marking a milestone along the long-term path of formalising Fermat's Last Theorem, Tate's thesis and the Langlands program within MATHLIB.

[1] https://github.com/smmercuri/adele-ring_locally-compact/tree/journal

Prior work [FF22] formalised the adele ring of a global field in Lean. Developments in MATHLIB since then enable us to formalise the completion of a number field at its infinite places, which we use to give a new formalisation of the infinite adele ring of a number field. With the help of recent work in local fields, [FFN24], we also provide important and necessary topological results, such as the proof that the local ring of integers is compact. We note that our work depends on the result, from [FFN24], that the v-adic ring of integers $\mathcal{O}_v$ for number fields has a finite residue field. At the time of writing, this result is being upstreamed to MATHLIB[2,3]. To avoid duplicating work, we assume this result in our project and leave the proof as a `sorry`.

We provide some mathematical preliminaries in Section 2, involving the definitions of absolute values, discrete valuations, number fields, and locally compact topological spaces, as well as completions of uniform spaces. In Section 3, we discuss families of induced absolute values on a field and how we handle multiple topological instances coming from them in Lean. Section 4 concerns the formalisation of the adele ring of a number field; in particular, the new formalisations of the completion of a number field at an infinite place and the infinite adele ring are detailed in Section 4.1. We then describe the informal and formal proof that the adele ring is locally compact, along with other topological results, in Section 5. Section 6 concludes with a discussion around implementation details.

2 Preliminaries

2.1 Absolute values and discrete valuations

While absolute values and discrete valuations are often unified into a single treatment, we separate them here due to their distinction within MATHLIB. See [CF67, Chapter 2] and [Ser79, Chapter 1] for details on the theory of absolute values and discrete valuations respectively, and [FFN24, Section 2] for details of integer-valued multiplicative valuations in MATHLIB.

***Definition* 2.1** (`AbsoluteValue`)**.** *Let K be a field. We say that the function $|\cdot| : K \to \mathbb{R}_{\geq 0}$ is an* absolute value *on K if, for all $x, y \in K$, we have:*

(1) $|x| = 0$ *if and only if* $x = 0$*;*
(2) $|xy| = |x||y|$*;*
(3) $|x + y| \leq |x| + |y|$ *(the* triangle inequality*).*

[2] https://github.com/leanprover-community/mathlib4/pull/26537
[3] https://github.com/leanprover-community/mathlib4/pull/26538

If the ultrametric inequality*,*

(4) $|x+y| \leq \max(|x|,|y|)$*,*

holds over the triangle inequality, the absolute value is non-Archimedean*; else, it is* Archimedean*.*

Discrete *additive* valuations on K are a subclass of group homomorphisms $K^\times \to (\mathbb{Z},+)$ that are extended to K as follows.

***Definition* 2.2** (`AddValuation`)**.** *The function* $a : K \to \mathbb{Z} \cup \{\infty\}$ *is a* discrete additive valuation *on* K *if, for all* $x, y \in K$*, we have:*

(1) $a(x) = \infty$ *if and only if* $x = 0$*;*
(2) $a(xy) = a(x) + a(y)$*;*
(3) $a(x+y) \geq \min(a(x), a(y))$*.*

Example 2.3. If $K = \mathbb{Q}$, then the additive p-adic valuation a_p (defined on an integer as the largest power of p dividing that integer; extended to $\mathbb{Q}$ by $a_p(a/b) := a_p(a) - a_p(b)$) is an example of a discrete additive valuation.

We can construct a non-Archimedean absolute value from a discrete additive valuation a by defining $|x|_a = c^{-a(x)}$ if $x \neq 0$ and $|0|_a = 0$, for some chosen real number $c > 1$. The p-adic absolute value $|x|_p = p^{-a_p(x)}$ is an example of this construction when $K = \mathbb{Q}$.

Discrete valuations in the literature are typically defined as additive valuations. In MATHLIB, discrete *multiplicative* valuations have a more developed API than their additive counterparts. These are abstractions of the absolute values arising from discrete additive valuations that do not require a choice of the base c. In MATHLIB, any abstract additive structure can be turned into a corresponding abstract multiplicative structure, which is in bijection with the original additive type via the inverse morphisms `ofAdd` and `toAdd`. For example, the additive group of integers $(\mathbb{Z},+)$ bijects with the type $\mathbb{Z}_m$, which can be thought of as the multiplicative group $\{x^n \mid n \in \mathbb{Z}\}$, where x is an abstract symbol, as:

$$\texttt{ofAdd} : (\mathbb{Z},+) \to (\mathbb{Z}_m,\times);$$
$$\texttt{toAdd} : (\mathbb{Z}_m,\times) \to (\mathbb{Z},+).$$

In particular, these maps preserve respective units, generators and preorders, so $\texttt{ofAdd}(0) = 1$, $\texttt{ofAdd}(1)$ generates $(\mathbb{Z}_m,\times)$, and $\texttt{ofAdd}(x) \leq \texttt{ofAdd}(y)$ if and only if $x \leq y$. The map `ofAdd` can be thought of as $n \mapsto x^n$, where x is a symbol. To transfer infinite additive values to

the multiplicative setting we add a zero term to obtain the type $\mathbb{Z}_{m0}$, which extends the multiplicative structure and the preorder of $\mathbb{Z}_m$.

Definition 2.4 (`Valuation`)**.** *The function $v : K \to \mathbb{Z}_{m0}$ is a* discrete multiplicative valuation *on K if, for all $x, y \in K$, we have:*

(1) $v(0) = 0$;
(2) $v(1) = 1$;
(3) $v(xy) = v(x)v(y)$;
(4) $v(x + y) \leq \max(v(x), v(y))$.

We emphasise that a discrete multiplicative valuation is not the same as the absolute value $|\cdot|_a$ constructed from a discrete additive valuation a. The former only takes values in $\mathbb{Z}_{m0}$, while the latter does not; the latter depends on a choice of base c, while the former does not. However, each additive valuation a also gives rise to a multiplicative valuation v by defining $v(x) = \texttt{ofAdd}(-a(x))$ for $x \neq 0$ and $v(0) = 0$.

Example 2.5. If $K = \mathbb{Q}$, then the p-adic valuation v_p is defined by $v_p(x) = \texttt{ofAdd}(-a_p(x))$ if $x \neq 0$. We see that x is a p-adic unit if and only if $v_p(x) = \texttt{ofAdd}(0) = 1$, and $v_p(x) \leq 1$ if and only if $a_p(x) \geq 0$.

Definition 2.6. *Two absolute values (resp. discrete multiplicative valuations) $|\cdot|_1$ and $|\cdot|_2$ (resp. v_1 and v_2) on K are* equivalent *if $|\cdot|_1 = |\cdot|_2^\alpha$ (resp. $v_1 = v_2^\alpha$) for some $\alpha \in \mathbb{R}_{>0}$.*

Equivalent valuations define the same uniform structures on K and lead to the same completions of K, so we typically care more about equivalence classes of valuations.

Definition 2.7. *An* infinite place *of a field K is an equivalence class of Archimedean absolute values on K* (`NumberField.InfinitePlace`)*. A* finite place *of K is an equivalence class of discrete multiplicative valuations on K.*

Remark 2.8. Note that both the predicate that two absolute values are equivalent and the type of finite places of a number field have since been included in MATHLIB[4,5]. Instead, we worked directly with valuations coming from distinct non-zero prime ideals.

[4] https://github.com/leanprover-community/mathlib4/pull/20362
[5] https://github.com/leanprover-community/mathlib4/pull/19667

2.2 Number fields

A number field K is a finite-degree field extension of $\mathbb{Q}$, and its ring of integers $\mathcal{O}_K$ is the integral closure of $\mathbb{Z}$ inside K. If $\mathcal{O}_K$ is not a unique factorisation domain, then the fundamental theorem of arithmetic does not hold at the level of elements, however it does hold at the level of ideals. That is, any non-zero ideal of $\mathcal{O}_K$ can be written as a finite product of non-zero prime ideals, unique up to reordering of the prime ideals.

***Definition* 2.9** (𝔭-adic valuation, `IsDedekindDomain.HeightOneSpectrum.valuation`)**.** *The* additive 𝔭-adic valuation $a_\mathfrak{p}(x)$ *on a non-zero prime ideal* $\mathfrak{p}$ *of* $\mathcal{O}_K$ *is the largest power of* $\mathfrak{p}$ *appearing in the factorisation of* $x\mathcal{O}_K$ *into prime ideals. This extends to a discrete additive valuation on* K *by defining* $a_\mathfrak{p}(a/b) = a_\mathfrak{p}(a) - a_\mathfrak{p}(b)$ *and* $a_\mathfrak{p}(0) = \infty$*. The* 𝔭-adic valuation $v_\mathfrak{p}$ *is the multiplicative valuation defined as* $v_\mathfrak{p}(x) = \texttt{ofAdd}(-a_\mathfrak{p}(x))$ *if* $x \neq 0$ *and* $v_\mathfrak{p}(0) = 0$*.*

2.3 Uniform spaces and their completions

While topological spaces abstract the notion of continuity, *uniform spaces*, first introduced by [Wei38], provide the appropriate abstraction of *uniform* continuity. We describe some brief relevant theory here; for more information, we refer the reader to [BCM20, Sections 2, 5]. A uniform space $(X, \mathcal{U})$ consists of a set X and a *uniform structure* $\mathcal{U}$, which is a filter on $X \times X$ satisfying certain properties. Uniform continuity of a map $f : (X, \mathcal{U}_X) \to (Y, \mathcal{U}_Y)$ between uniform spaces is understood via the condition that every set in $\mathcal{U}_X$ is also in $(f \times f)^*\mathcal{U}_Y$, where $(f \times f)^*\mathcal{U}_Y$ is the *pullback* filter generated by preimages of elements in $\mathcal{U}_Y$ under $f \times f$.

Uniform spaces come with a powerful completion operation, which makes use of the unified notion of limits that filters represent. Much like metric spaces can be completed by adding limits of Cauchy sequences, there is an operation sending a uniform space $(X, \mathcal{U})$ to a complete separated uniform space $(\widehat{X}, \widehat{\mathcal{U}})$ by adding limits of *Cauchy filters*, [BCM20]. This completion operator satisfies the universal property of [Bou66, Theorem II.3.7.3, Definition II.3.7.4] (see also [BCM20, Theorem 5.1]), hence it defines a functor from the category of all uniform spaces to the category of complete separated uniform spaces. The universal property of the uniform space completion functor means that it represents an equivalence class of *abstract completion operators* as follows.

***Definition* 2.10** ([BCM20], `AbstractCompletion`)**.** *We say that* $(Y, \iota : X \to Y)$ *is an* abstract completion *of the uniform space* X *if* Y *is a complete separated uniform space,* ι *has dense image and* ι *is uniformly continuous.*

Theorem 2.11 ([BCM20], `AbstractCompletion.compareEquiv`)**.** *Any two abstract completions* $(Y_i, \iota_i : X \to Y_i)$*, for* $i = 1, 2$*, of a uniform space* X *are isomorphic as uniform spaces.*

These properties make the uniform space completion operation ideal for formalising the completions of a range of objects, including the number fields that we consider within this paper. This theory is contained in MATHLIB within the `UniformSpace.Completion` API, much of which is inherited from the API of the `AbstractCompletion` structure.

A field K can have various sources of uniform structures. An absolute value $|\cdot|$ on K determines a uniform structure through the filter generated by the sets $\{(x, y) \mid |x - y| < \varepsilon\}$, where $\varepsilon > 0$. Discrete multiplicative valuations v similarly define uniform spaces $(K, \mathcal{U}_v)$. These uniform structures determine the completion K_v of a field at each place v via the uniform space completion functor.

Example 2.12. (1) The uniform structure $\mathcal{U}_\infty$ coming from the usual absolute value $|\cdot|_\infty$ on $\mathbb{Q}$ defines a uniform space $(\mathbb{Q}, \mathcal{U}_\infty)$, whose completion is the field $\mathbb{R}$ of real numbers.

(2) The uniform structure $\mathcal{U}_{v_p}$ coming from the multiplicative p-adic valuation defines a uniform space $(\mathbb{Q}, \mathcal{U}_{v_p})$, whose completion is the field $\mathbb{Q}_p$ of p-adic numbers. The subring $\mathbb{Z}$ of $\mathbb{Q}$ completes to the p-adic ring of integers $\mathbb{Z}_p := \{x \in \mathbb{Q}_p \mid v_p(x) \leq 1\}$.

2.4 Locally compact spaces

There are many characterisations of a topological space being locally compact. The following corresponds to the implementation found in MATHLIB.

***Definition* 2.13** (`LocallyCompactSpace`)**.** *A topological space* X *is* locally compact *if, for each neighbourhood* N *of each* $x \in X$*, there exists a compact neighbourhood* $S \subseteq N$ *of* x*.*

It is well known that closed subspaces of locally compact spaces, finite products of locally compact spaces, and infinite products of compact spaces are all locally compact.

3 Induced absolute values and their pullback uniform structures

Throughout Section 3 the following variables are in scope.

```
variable {K L : Type*} [Field K] [NormedField L] (v : AbsoluteValue K ℝ)
```

3.1 Handling multiple instances on a type using dependent type synonyms

As described in Section 2, there may be multiple distinct `UniformSpace` instances on a number field coming from its infinite places. However, Lean's type class inference system cannot automatically resolve multiple instances of the same class on a single type. If we assign multiple `UniformSpace` instances directly to the type `K`, we will then be forced to manually specify the desired instance through the use of `@` or local `let` declarations.

An alternative approach to handling this issue is through the use of type synonyms. Type synonyms simply rename a type, however they also allow us to insert dependencies. This provides a mechanism through which the type class inference system is able to resolve multiple dependent instances on a type. For a general semiring `R` and ordered semiring `S`, we define the type synonym `WithAbs` of `R`, which depends on an `S`-valued absolute value on `R`.

```
def WithAbs {R S} [Semiring R] [OrderedSemiring S] : AbsoluteValue R S → Type _ :=
 fun _ => R
```

In particular, given a *real* absolute value `v` on a field `K`, we assign a `NormedField` instance to `K` by assigning it to its type synonym `WithAbs v`.

```
instance normedField : NormedField (WithAbs v) where ...
```

Now there is only a *single* `NormedField` instance on each `WithAbs v`, enabling the type class inference system to automatically resolve the relevant `NormedField` and its parent `UniformSpace` instance on `K` coming from the absolute value `v`. The completion of a field with respect to an absolute value can then be given by applying `UniformSpace.Completion`.

```
abbrev AbsoluteValue.Completion := UniformSpace.Completion (WithAbs v)
```

See Section 6.1 for a comparison of the dependent type synonym approach to other approaches for handling the multiple instance problem.

3.2 Induced absolute values

A family of uniform structures on a field K can be generated through field embeddings $K \hookrightarrow L$ of K into a field L, where L has an absolute value $|\cdot|_L : L \to \mathbb{R}$ as follows.

***Definition* 3.1.** *Let K and L be fields and let $|\cdot|_L : L \to \mathbb{R}$ be an absolute value on L. If $\sigma : K \hookrightarrow L$ is a field embedding, then the* σ-induced *absolute value $|\cdot|_\sigma : K \to \mathbb{R}$ is defined by $|x|_\sigma = |\sigma(x)|_L$ for all $x \in K$.*

The uniform structure $\mathcal{U}_\sigma$ on K given by a $(\sigma : K \hookrightarrow L)$-induced absolute value $|\cdot|_\sigma$ is precisely the pullback, $(\sigma \times \sigma)^* \mathcal{U}_L$, of the uniform structure $\mathcal{U}_L$ given by the absolute value $|\cdot|_L$. In MATHLIB, this property is encoded in the `UniformInducing` definition.

```
theorem WithAbs.uniformInducing_of_comp {f : WithAbs v →+* L}
  (h : ∀ x, ‖f x‖ = v x) : UniformInducing f := ...
```

3.3 Pullbacks preserve the completable topological field property

In general, the completion $\widehat{K}$ of a separated uniform space $(K, \mathcal{U})$ may not be a field, even if K is a field. The necessary and sufficient condition required to guarantee this is that $(K, \mathcal{U})$ is a *completable topological field*, given in [Bou66, Theorem III.6.8.7] and also [BCM20, Theorem 5.4], which requires the inverse map $x \mapsto x^{-1}$ on K to preserve the Cauchy property of filters which do not have a cluster point at zero. The type class `CompletableTopField` encodes this property within MATHLIB.

Let $(L, \mathcal{U}_L)$ be a completable topological field. The pullback $(\sigma \times \sigma)^* \mathcal{U}_L$ uniform structure on K under a field embedding $\sigma : K \hookrightarrow L$ defines a completable topological field.

```
theorem UniformInducing.completableTopField {K L : Type*} [Field L]
  [UniformSpace L] [CompletableTopField L] [Field K] [UniformSpace K]
  [T0Space K] {f : K →+* L} (hf : UniformInducing f) :
  CompletableTopField K := ...
```

Since the uniform structure of a $(\sigma : K \hookrightarrow L)$-induced absolute value of a field K coincides with the pullback $(\sigma \times \sigma)^* \mathcal{U}_L$, we see that $(K, \mathcal{U}_\sigma)$ is a completable topological field. The completion of K with respect to a σ-induced absolute value is therefore a field.

4 The adele ring of a number field

Ostrowski's theorem [Ost16] states that the only infinite place of $\mathbb{Q}$ is represented by the usual absolute value $|\cdot|_\infty$ and all the finite places are represented by the p-adic valuations v_p. This result has been formalised and is part of a later version of MATHLIB[6]. The *infinite adele ring* $\mathbb{A}_{\mathbb{Q},\infty}$ of $\mathbb{Q}$ is the completion $\mathbb{R}$ of $\mathbb{Q}$ at the single infinite place, and the *finite adele ring* $\mathbb{A}_{\mathbb{Q},f}$ is the restricted product of the completions of $\mathbb{Q}$ over primes p,

$$\mathbb{A}_{\mathbb{Q},f} := \left\{ (x_p)_p \in \prod_p \mathbb{Q}_p \;\middle|\; x_p \in \mathbb{Z}_p \text{ for all but finitely many } p \right\}. \tag{4.1}$$

The *adele ring* of $\mathbb{Q}$ is the product of the infinite and finite adele rings, $\mathbb{A}_\mathbb{Q} := \mathbb{A}_{\mathbb{Q},\infty} \times \mathbb{A}_{\mathbb{Q},f}$. The story is similar for general number fields, with the infinite places coming from the complex absolute value $|\cdot|_\mathbb{C}$ and the finite places being represented by $\mathfrak{p}$-adic valuations of non-zero prime ideals $\mathfrak{p}$ of $\mathcal{O}_K$.

[6] https://github.com/leanprover-community/mathlib4/pull/17138

4.1 The infinite adele ring of a number field

Throughout Section 4.1 the following variables are in scope.

```
variable (K : Type*) [Field K] [NumberField K] (v : NumberField.InfinitePlace K)
```

4.1.1 Infinite places of a number field

There is only a single infinite place of $\mathbb{Q}$ because there is only a single field embedding of $\mathbb{Q}$ into $\mathbb{C}$. In general, number fields K have a finite number of distinct embeddings. For example, the embeddings of $K = \mathbb{Q}(\alpha)$ correspond to the Galois action permuting the roots of the minimal polynomial of α over $\mathbb{Q}$. Distinct embeddings $\sigma : K \hookrightarrow \mathbb{C}$ generate non-equivalent Archimedean absolute values on K via the σ-induced absolute values $|x|_\sigma := |\sigma(x)|_{\mathbb{C}}$. These define all the infinite places of a number field, the set of which is denoted by $\Sigma_{K,\infty}$.

The type `NumberField.InfinitePlace` in MATHLIB[7] encodes the infinite places of a number field K. Each term `v` consists of an absolute value `v.val : AbsoluteValue K ℝ`, and a proof (`‖v.embedding x‖ = v.val x` for all `x : K`) that `v.val` is induced by some `v.embedding : K →+* ℂ`.

4.1.2 Completing a number field at an infinite place

Infinite places $v \in \Sigma_{K,\infty}$ define distinct uniform structures $\mathcal{U}_v$ on K through their associated absolute values and we can therefore complete K at v to obtain K_v, as described in Section 2.3. Formally, we do this using `AbsoluteValue.Completion` of Section 3.1.

```
abbrev NumberField.InfinitePlace.Completion := v.val.Completion
```

Because these absolute values are $(\sigma : K \hookrightarrow \mathbb{C})$-induced, we have that K is a completable topological field with respect to each $\mathcal{U}_v$ (Section 3.3), and so the completion K_v is a field. Moreover, the absolute value on K extends to K_v to yield a `NormedField` instance.

```
instance : NormedField v.Completion :=
 letI := (WithAbs.uniformInducing_of_comp v.norm_embedding_eq).completableTopField
 UniformSpace.Completion.instNormedFieldOfCompletableTopField (WithAbs v.val)
```

4.1.3 The infinite adele ring

***Definition* 4.1** (`NumberField.InfiniteAdeleRing`)**.** *The infinite adele ring $\mathbb{A}_{K,\infty}$ of a number field K is the product topological ring over all infinite place completions of K:*

$$\mathbb{A}_{K,\infty} := \prod_{v \in \Sigma_{K,\infty}} K_v. \tag{4.2}$$

[7] https://github.com/leanprover-community/mathlib3/pull/17844

```
def NumberField.InfiniteAdeleRing := (v : InfinitePlace K) → v.Completion

instance : TopologicalRing (InfiniteAdeleRing K) := Pi.instTopologicalRing
```

4.2 The finite adele ring of a number field

The finite adele ring was formalised in MATHLIB, following [FF22], for any Dedekind domain R of Krull dimension one and its field of fractions K, by taking the restricted product over the non-zero prime spectrum $\mathrm{Spec}(R)$ of R. Formally, we work in the same generality. However, for simplicity our informal discussion restricts to $R = \mathcal{O}_K$, as required by our main results on number fields. Throughout Section 4.2 the following variables are in scope.

```
variable (R : Type*) [CommRing R] [IsDomain R] [IsDedekindDomain R]
 (K : Type*) [Field K] [Algebra R K] [IsFractionRing R K]
```

4.2.1 Finite places of a number field

Discrete valuations of global fields and their completions at finite places were previously formalised in [FF22; FFN24]. The set of all finite places of a number field K, denoted by $\Sigma_{K,f}$, is indexed by the non-zero prime ideals $\mathfrak{p} \in \mathrm{Spec}(\mathcal{O}_K)$ through their $\mathfrak{p}$-adic valuations (Definition 2.9). The MATHLIB structure `Valuation K` $\mathbb{Z}_{m0}$ represents discrete multiplicative valuations of K. The typing `v : IsDedekindDomain.HeightOneSpectrum (𝓞 K)` represents the relation $\mathfrak{p} \in \mathrm{Spec}(\mathcal{O}_K)$ and its multiplicative valuation is `v.valuation : Valuation K` $\mathbb{Z}_{m0}$. Moreover, each prime ideal term `v` defines a uniform structure on K, via `v.adicValued.toUniformSpace`, which is then used to formalise the v-adic completion K_v of K as `v.adicCompletion K`. The v-adic ring of integers, $\mathcal{O}_v := \{x \in K_v \mid v(x) \leq \texttt{ofAdd}(0)\}$, is the subring `v.adicCompletionIntegers K`, which is a discrete valuation ring with unique maximal ideal $\mathfrak{m}_v := \{x \in \mathcal{O}_v \mid v(x) < \texttt{ofAdd}(0)\}$. This is a principal ideal generated by a choice of uniformizer π, which is any element such that $v(\pi) = \texttt{ofAdd}(-1)$ (this corresponds to having additive valuation $a(\pi) = 1$). The formal maximal ideal of $\mathcal{O}_v$ is `Valued.maximalIdeal (v.adicCompletion K)`.

4.2.2 The finite adele ring

***Definition* 4.2** (`DedekindDomain.FiniteAdeleRing`)**.** *The* finite adele ring $\mathbb{A}_{K,f}$ *of a number field* K*,*

$$\mathbb{A}_{K,f} := \left\{ (x_v)_v \in \prod_{v \in \Sigma_{K,f}} K_v \;\middle|\; x_v \in \mathcal{O}_v \text{ for all but finitely many } v \right\}, \tag{4.3}$$

is the topological ring whose topology is generated by the basis $\left\{\prod_v q\mathcal{O}_v \;\middle|\; 0 \neq q \in \mathcal{O}_K\right\}$ *of neighbourhoods of zero.*

In the version of MATHLIB used in this project, the finite adele ring of any Dedekind domain `R` and its field of fractions `K` was introduced by the work of [FF22]. Its topological space structure was incorporated into MATHLIB in a later PR[8]; this makes use of the `SubmodulesRingBasis` formalism which constructs a `TopologicalSpace` instance from a basis of neighbourhoods of zero, which is compatible with the underlying ring structure. In other words, it also generates the appropriate `TopologicalRing` instance.

Remark 4.3. Open sets containing zero in the finite adele ring are generated by sets of the form $\prod_{v\in S} B_{r_v}(0) \times \prod_{v\notin S} \mathcal{O}_v$, where S is finite and $0 < r_v \in \mathbb{Z}$. The family of all such sets is parameterised by $0 \neq q \in K$ via $\prod_v q\mathcal{O}_v$. However, the neighbourhoods of zero filter is generated through a basis by upwards inclusion. Since $\mathcal{O}_v \subseteq q\mathcal{O}_v$ for any v dividing the denominator of q, it suffices to consider only $q \in \mathcal{O}_K$ in order to give a basis of the neighbourhoods of zero.

Remark 4.4. Since the completion of this project, both the definition and the topology of the finite adele ring in MATHLIB have seen a significant refactor to use the `RestrictedProduct` API[9,10]. The impact of this refactor on the local compactness of the finite adele ring is discussed in Remark 5.4.

4.3 The adele ring

***Definition* 4.5** (`NumberField.AdeleRing`)**.** *The adele ring* $\mathbb{A}_K$ *of a number field is the product topological ring of the infinite and the finite adele rings:*

$$\mathbb{A}_K := \mathbb{A}_{K,\infty} \times \mathbb{A}_{K,f}. \tag{4.4}$$

```
def NumberField.AdeleRing (K : Type*) [Field K] [NumberField K] :=
  InfiniteAdeleRing K × DedekindDomain.FiniteAdeleRing (𝒪 K) K
```

5 Local compactness of the adele ring of a number field

This section contains an informal and formal proof that the adele ring of a number field is locally compact, using [CF67, Chapter 2] as source. Full details of the formal proof can

[8] https://github.com/leanprover-community/mathlib4/pull/14176
[9] https://github.com/leanprover-community/mathlib4/pull/20021
[10] https://github.com/leanprover-community/mathlib4/pull/23542

be found in the source code. The informal proofs are written so as to align with the formal proofs.

5.1 Local compactness of the infinite adele ring

Throughout Section 5.1 the following variables are in scope.

```
variable {K L : Type*} [Field K] [NormedField L] [CompleteSpace L]
 {v : AbsoluteValue K ℝ} {f : WithAbs v →+* L}
```

We first show that the completion of a number field at an infinite place is locally compact. As in Sections 3 and 4, our strategy is to first formalise the local compactness in higher abstraction for fields with an associated induced absolute value, from which the result for number fields and infinite places can be derived.

Theorem 5.1. *Let K and L be fields, let $|\cdot|_L : L \to \mathbb{R}$ be an absolute value on L, and assume that L is locally compact and complete with respect to the topology induced by $|\cdot|_L$. If $\sigma : K \hookrightarrow L$ is a field embedding, then the completion of K with respect to the σ-induced absolute value $|\cdot|_\sigma$ is locally compact.*

Proof. Since the uniform structure given by $|\cdot|_\sigma$ is the pullback uniform structure, σ is uniformly continuous. Therefore σ extends to an embedding $\sigma_v : K_v \hookrightarrow L$ by the universal property of uniform completions. Further, σ and σ_v are both isometries; thus, K_v has a closed image under σ_v and so is locally compact. □

```
namespace AbsoluteValue.Completion

abbrev extensionEmbeddingOfComp (h : ∀ x, ‖f x‖ = v x) : v.Completion →+* L :=
 UniformSpace.Completion.extensionHom _
  (WithAbs.uniformInducing_of_comp h).uniformContinuous.continuous

theorem extensionEmbeddingOfComp_dist_eq (h : ∀ x, ‖f x‖ = v x)
  (x y : v.Completion) :
  dist (extensionEmbeddingOfComp h x) (extensionEmbeddingOfComp h y) =
   dist x y := by ...

theorem isometry_extensionEmbeddingOfComp (h : ∀ x, ‖f x‖ = v x) :
  Isometry (extensionEmbeddingOfComp h) :=
 Isometry.of_dist_eq <| extensionEmbeddingOfComp_dist_eq h
```

```
theorem closedEmbedding_extensionEmbeddingOfComp (h : ∀ x, ‖f x‖ = v x) :
  ClosedEmbedding (extensionEmbeddingOfComp h) :=
 (isometry_extensionEmbeddingOfComp h).closedEmbedding

theorem locallyCompactSpace [LocallyCompactSpace L] (h : ∀ x, ‖f x‖ = v x) :
  LocallyCompactSpace (v.Completion) :=
 (closedEmbedding_extensionEmbeddingOfComp h).locallyCompactSpace

end AbsoluteValue.Completion
```

Corollary 5.2. *The completion K_v of a number field K at $v \in \Sigma_{K,\infty}$ is locally compact.*

```
instance NumberField.InfinitePlace.Completion.locallyCompactSpace
  (v : InfinitePlace K) : LocallyCompactSpace (v.Completion) :=
 AbsoluteValue.Completion.locallyCompactSpace v.norm_embedding_eq
```

Theorem 5.3. *The infinite adele ring $\mathbb{A}_{K,\infty}$ of a number field K is locally compact.*

Proof. Since each completion K_v is locally compact by Corollary 5.2 then, as a finite product of locally compact spaces, so is $\mathbb{A}_{K,\infty}$. □

```
theorem NumberField.InfiniteAdeleRing.locallyCompactSpace [NumberField K] :
  LocallyCompactSpace (InfiniteAdeleRing K) :=
 Pi.locallyCompactSpace_of_finite
```

5.2 Local compactness of the finite adele ring

As a subring of an infinite product, it is not immediately clear that the finite adele ring is locally compact. However, each $\mathcal{O}_v$ is *compact*, and each finite adele is in $\mathcal{O}_v$ at infinitely-many places. This is the reason why the finite adele ring is locally compact. To prove this, we first show that $\mathcal{O}_v$ is compact and that K_v is locally compact in Section 5.2.1. Then in Sections 5.2.2 and 5.2.3 we show that the finite adele ring admits an open cover of locally compact *finite S-adele rings*.

Remark 5.4. Since the completion of this project, the finite adele ring has seen a significant refactor to use the `RestrictedProduct` API[11,12]. While the compactness results of Section 5.2.1

[11] https://github.com/leanprover-community/mathlib4/pull/20021
[12] https://github.com/leanprover-community/mathlib4/pull/23542

remain essential, the results from Sections 5.2.2 and 5.2.3 on the local compactness of the finite adele ring have now been generalised to a corresponding result on restricted products. Some of the auxiliary objects used in this section, such as `ProdAdicCompletions` and `FiniteIntegralAdeles`, have since been removed from MATHLIB. Nevertheless, we retain the formal S-adele argument here as a record of the first formalised proof of this fact.

Throughout Section 5.2 the following variables are in scope.

```
variable {R : Type*} [CommRing R] [IsDedekindDomain R]
 (K : Type*) [Field K] [Algebra R K] [IsFractionRing R K] [NumberField K]
 (v : HeightOneSpectrum R) (S : Finset (HeightOneSpectrum R))
```

5.2.1 Compactness of the v-adic ring of integers

Theorem 5.5. *The ring of integers $\mathcal{O}_v$ of the completion K_v of a number field K at $v \in \Sigma_{K,f}$ is compact.*

Proof. We show equivalently that $\mathcal{O}_v$ is complete and totally bounded. As a closed subset of the complete space K_v, $\mathcal{O}_v$ is complete. It is totally bounded if, for any $\gamma \in \mathbb{Z}_m$, there exists a finite cover of $\mathcal{O}_v$ of open balls $B_\gamma(t) := \{x \mid v(x-t) < \gamma\}$ of radius γ. It suffices to check this for $\gamma \leq 1$, in which case we take the finitely-many representatives t_i of $\mathcal{O}_v/\mathfrak{m}_v^{-\mathtt{toAdd}(\gamma)+1}$. The balls $B_\gamma(t_i)$ then cover $\mathcal{O}_v$. □

```
namespace IsDedekindDomain.HeightOneSpectrum.adicCompletionIntegers

theorem isClosed : IsClosed (v.adicCompletionIntegers K).carrier := ...

theorem totallyBounded : TotallyBounded (v.adicCompletionIntegers K).carrier := ...

theorem isCompact : IsCompact (v.adicCompletionIntegers K).carrier :=
 isCompact_iff_totallyBounded_isComplete.2
  ⟨totallyBounded K v, (isClosed K v).isComplete⟩

instance compactSpace : CompactSpace (v.adicCompletionIntegers K) :=
 isCompact_iff_compactSpace.1 <| isCompact K v
```

The proof of Theorem 5.5 requires that the residue field $\mathcal{O}_v/\mathfrak{m}_v$ is *finite*. As described in the introduction, this has been formalised elsewhere, [FFN24], and is currently in the process of being upstreamed to MATHLIB[13,14]. We assume the statement with a `sorry` proof in

[13] https://github.com/leanprover-community/mathlib4/pull/26537
[14] https://github.com/leanprover-community/mathlib4/pull/26538

our work to avoid duplication. However, we prove under its assumption that $\mathcal{O}_v/\mathfrak{m}_v^n$ is finite for any integer $n \geq 0$. To do this we define the finite-coefficient map `toFiniteCoeffs n hπ` sending $x \in \mathcal{O}_v/\mathfrak{m}_v^n$ to an n-tuple $(x_1, ..., x_n)$ of v-adic digits in its π-adic expansion, where π is a uniformizer. This gives an injective function $\mathcal{O}_v/\mathfrak{m}_v^n \to (\mathcal{O}_v/\mathfrak{m}_v)^n$.

```
theorem toFiniteCoeffs_injective {π : v.adicCompletionIntegers K} (n : ℕ)
  (hπ : IsUniformizer π.val) : (toFiniteCoeffs n hπ).Injective := ...

instance quotient_maximalIdeal_pow_finite {π : v.adicCompletionIntegers K} (n : ℕ)
  (hπ : IsUniformizer π.val) :
  Finite (v.adicCompletionIntegers K /
   (Valued.maximalIdeal (v.adicCompletion K)) ^ n) :=
 Finite.of_injective _ (toFiniteCoeffs_injective n hπ)
```

Theorem 5.6. *The completion K_v of a number field at $v \in \Sigma_{K,f}$ is locally compact.*

Proof. It is enough to show that 0 has a compact neighbourhood because we can translate and dilate this to be contained in neighbourhoods around other points. The neighbourhood $B_1(0)$ of 0 is a closed subspace of the compact space $\mathcal{O}_v$, so it is compact. □

```
theorem adicCompletion.isCompact_nhds_zero {γ : ℤₘ₀ˣ} (hγ : γ ≤ 1) :
  IsCompact { y : v.adicCompletion K | Valued.v y < γ } :=
 (isCompact K v).of_isClosed_subset (isClosed_nhds_zero K v γ)
  <| fun _ hx => le_of_lt (lt_of_lt_of_le (Set.mem_setOf.1 hx) hγ)

instance adicCompletion.locallyCompactSpace :
  LocallyCompactSpace (v.adicCompletion K) :=
 (isCompact_nhds_zero K v le_rfl).locallyCompactSpace_of_mem_nhds_of_addGroup
  <| (hasBasis_nhds_zero K v).mem_of_mem le_rfl
```

5.2.2 The finite S-adele ring

Associated to any element x of the finite adele ring is its *support* – a finite set of places $S_x \subseteq \Sigma_{K,f}$ for which $x_v \in \mathcal{O}_v$ if and only if $v \notin S_x$. It is the dependence of the set S_x on x that makes the local compactness of the finite adele ring difficult to view. On the other hand, if we fix a finite set $S \subseteq \Sigma_{K,f}$, and consider only the finite adeles x for which $S_x \subseteq S$, then this is an easier space to understand as locally compact. These are the finite S-adeles.

***Definition* 5.7** (DedekindDomain.FinsetAdeleRing)**.** *Let $S \subseteq \Sigma_{K,f}$ be a finite set of places of a number field K. The finite S-adele ring $\mathbb{A}_{K,S,f}$ is the topological ring defined as*

$$\mathbb{A}_{K,S,f} := \left\{ (x_v)_v \in \prod_{v \in \Sigma_{K,f}} K_v \;\middle|\; x_v \in \mathcal{O}_v \text{ for all } v \notin S \right\}.$$

Mathematically, $\mathbb{A}_{K,S,f}$ is given the subspace topology of $\prod_v K_v$, but it can also be viewed as a subspace of $\mathbb{A}_{K,f}$, so it has two immediate sources of topologies. These define the *same* topology on $\mathbb{A}_{K,S,f}$, which is a crucial step in the overall proof that $\mathbb{A}_{K,f}$ is locally compact.

We formalise $\mathbb{A}_{K,S,f}$ as a subtype of $\prod_v K_v$, distinct from $\mathbb{A}_{K,f}$. It is given the subspace topology of $\prod_v K_v$. It does not inherit the subspace topology of $\mathbb{A}_{K,f}$, but we prove in Section 5.2.3 that the $\mathbb{A}_{K,f}$-induced topology matches the inherited $\prod_v K_v$-subspace topology.

```
namespace DedekindDomain

def IsFinsetAdele (x : ProdAdicCompletions R K) :=
  ∀ v ∉ S, x v ∈ v.adicCompletionIntegers K

def FinsetAdeleRing := {x : ProdAdicCompletions R K // IsFinsetAdele S x}
```

We construct a `Subring (ProdAdicCompletions R K)` term `FinsetAdeleRing.subring R K`, with carrier `{x | IsFinsetAdele S x}`. This is used to infer the subspace `TopologicalRing` instance on `FinsetAdeleRing R K`. Defining `FinsetAdeleRing` as a type and then constructing a `Subring` term separately is modelled after PRs[15,16] which converted the ring of integers of a number field and the finite adele ring to types from `Subalgebra` and `Subring` terms respectively. The former PR demonstrated significant performance improvements in MATHLIB, with files seeing up to 71.9% faster build time.

Theorem 5.8. *The finite S-adele ring $\mathbb{A}_{K,S,f}$ of a number field K is locally compact.*

Proof. The finite S-adele ring is the product

$$\prod_{v \in S} K_v \times \prod_{v \notin S} \mathcal{O}_v,$$

whose first factor is locally compact as a finite product of locally compact spaces, and whose second factor is locally compact as an infinite product of compact spaces. □

[15] https://github.com/leanprover-community/mathlib4/pull/12386
[16] https://github.com/leanprover-community/mathlib4/pull/13021

The above proof hides some details that are trivial mathematically, but require care in formalisation. This comes down to two concerns: (1) how we formalise the space $\prod_{v\in S} K_v \times \prod_{v\notin S} \mathcal{O}_v$; (2) the sense in which we identify $\mathbb{A}_{K,S,f}$ and $\prod_{v\in S} K_v \times \prod_{v\notin S} \mathcal{O}_v$.

For (1), $\prod_{v\in S} K_v \times \prod_{v\notin S} \mathcal{O}_v$ can be formalised as a type, given a product topological space instance, which we denote mathematically also by $\prod_{v\in S} K_v \times \prod_{v\notin S} \mathcal{O}_v$. Or, it can be a subtype of $\prod_{v\in S} K_v \times \prod_{v\notin S} K_v$, given a `Subtype` topological space instance, which we denote mathematically by $\widehat{\mathcal{O}}_S$. We shall use both, because the former is really the space whose local compactness is deduced in the proof of Theorem 5.8 above, but the latter simplifies the transfer of this local compactness to $\mathbb{A}_{K,S,f}$ considerably. The spaces $\prod_{v\in S} K_v \times \prod_{v\notin S} K_v$, $\prod_{v\in S} K_v \times \prod_{v\notin S} \mathcal{O}_v$ and $\widehat{\mathcal{O}}_S$ are formalised respectively as follows.

```
def ProdAdicCompletions.FinsetProd := ((v : S) → v.val.adicCompletion K) ×
 ((v : {v // v ∉ S}) → v.val.adicCompletion K)

def FinsetIntegralAdeles := ((v : S) → v.val.adicCompletion K) ×
 ((v : {v // v ∉ S}) → v.val.adicCompletionIntegers K)

def FinsetIntegralAdeles.Subtype :=
 {x : FinsetProd R K S // ∀ v, x.2 v ∈ v.val.adicCompletionIntegers K}
```

The identification concern of (2) can then be achieved through composing two homeomorphisms t_1 and t_2, as illustrated in the commutative diagram of Figure 1. Here, $p : \prod_v K_v \to$ `Prop`; $x \mapsto \forall v \notin S, x_v \in \mathcal{O}_v$ is the subtype condition defining $\mathbb{A}_{K,S,f}$ and $q : (x_1, x_2) \mapsto \forall v \notin S, (x_2)_v \in \mathcal{O}_v$ is the subtype condition defining $\widehat{\mathcal{O}}_S$. The *partial function* $\langle \cdot, p\rangle : \prod_v K_v \rightharpoonup \mathbb{A}_{K,S,f}$ is the function defined on each $x \in \prod_v K_v$ satisfying $p(x)$, sending it to the corresponding element in the subtype $\mathbb{A}_{K,S,f}$, and $\langle \cdot, q\rangle$ is defined analogously. The homeomorphism t_π is given in MATHLIB as `Homeomorph.piEquivPiSubtypeProd`.

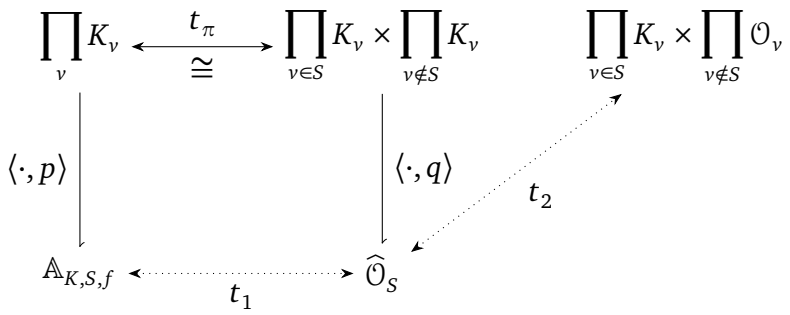

Figure 1: Type relationships in the formal proof for the local compactness of $\mathbb{A}_{K,S,f}$.

Given the formalisations of the local compactness of K_v and the compactness of $\mathcal{O}_v$ in Section 5.2.1, the local compactness of $\prod_v \in K_v \times \prod_{v \notin S} \mathcal{O}_v$ can be easily inferred.

```
instance FinsetIntegralAdeles.locallyCompactSpace :
  LocallyCompactSpace (FinsetIntegralAdeles R K S) :=
 Prod.locallyCompactSpace _ _
```

The homeomorphism t_2 is defined by `Prod.mk` and `Subtype.mk` maps, whose continuity is straightforward to show. This then gives the local compactness of $\widehat{\mathcal{O}}_S$.

```
def FinsetIntegralAdeles.subtypeHomeomorph :
  Subtype R K S ≃ₜ FinsetIntegralAdeles R K S where ...


instance FinsetIntegralAdeles.locallyCompactSpaceSubtype :
  LocallyCompactSpace (Subtype R K S) :=
 (subtypeHomeomorph R K S).closedEmbedding.locallyCompactSpace
```

It is easy to see that $p = q \circ t_\pi$ on defined domains, so the homeomorphism t_1 can be lifted from t_π using `Homeomorph.subtype` and used to infer that $\mathbb{A}_{K,S,f}$ is locally compact.

```
namespace FinsetAdeleRing

def homeomorphSubtype : FinsetAdeleRing R K S ≃ₜ FinsetIntegralAdeles.Subtype R K S :=
 (Homeomorph.piEquivPiSubtypeProd _ _).subtype <| fun _ =>
  ⟨fun hx v => hx v.1 v.2, fun hx v hv => hx ⟨v, hv⟩⟩

instance locallyCompactSpace : LocallyCompactSpace (FinsetAdeleRing R K S) :=
 (homeomorphSubtype R K S).closedEmbedding.locallyCompactSpace
```

5.2.3 Using the finite S-adele rings to cover the finite adele ring

Theorem 5.9. *The finite adele ring $\mathbb{A}_{K,f}$ of a number field K is locally compact.*

Proof. Let $x \in \mathbb{A}_{K,f}$ and N be a neighbourhood of x. We have $x \in \mathbb{A}_{K,S_x,f}$, where S_x is the set of finite places for which $x \notin \mathcal{O}_v$. By the definition of the topology on the finite adele ring, $\mathbb{A}_{K,S_x,f}$ is a neighbourhood of x. Moreover, it is locally compact by Theorem 5.8. The neighbourhood $N \cap \mathbb{A}_{K,S_x,f}$ of x inside the finite S-adele ring therefore contains a compact neighbourhood M of x. The inclusion map $\mathbb{A}_{K,S_x,f} \hookrightarrow \mathbb{A}_{K,f}$ sends neighbourhoods to neighbourhoods and is continuous open, so the image of M into $\mathbb{A}_{K,f}$ is a compact neighbourhood of x in $\mathbb{A}_{K,f}$. □

Remark 5.10. The proof in this section shows local compactness by direct satisfaction of Definition 2.13. In our code, we also provide a simplified proof which takes advantage of

the fact that, in a topological ring, we need only show that 0 has a compact neighbourhood, which is given by $\prod_v \mathcal{O}_v$ in this case.

The subtlety here is that, in Theorem 5.8, we showed that the finite S-adele ring with the subspace topology of $\prod_v K_v$ is locally compact, whereas in the above proof, we are viewing it as a subspace of $\mathbb{A}_{K,f}$. The key point is that these topologies coincide. The $\mathbb{A}_{K,f}$-subspace topology is defined by inducing through the embedding $\iota(S) : \mathbb{A}_{K,S,f} \hookrightarrow \mathbb{A}_{K,f}$.

```
instance : Algebra (FinsetAdeleRing R K S) (FiniteAdeleRing R K) where ...

local notation "ι(" S ")" => algebraMap (FinsetAdeleRing R K S) (FiniteAdeleRing R K)

theorem algebraMap_injective : Function.Injective ι(S) := ...
```

As constructed, `FinsetAdeleRing R K S` has the `ProdAdicCompletions R K` subspace topology. In MATHLIB, the property that the topology induced through a function coincides with a space's given topology is encoded in the `Inducing` structure. The map $\iota(S)$ is `Inducing` if and only if the images of neighbourhoods in $\mathbb{A}_{K,S,f}$ and preimages of neighbourhoods in $\mathbb{A}_{K,f}$ remain neighbourhoods under $\iota(S)$.

```
theorem algebraMap_image_mem_nhds (x : FinsetAdeleRing R K S)
  {U : Set (FinsetAdeleRing R K S)} (h : U ∈ nhds x) :
  ι(S) '' U ∈ nhds (ι(S) x) := by ...

theorem mem_nhds_comap_algebraMap (x : FinsetAdeleRing R K S)
  {U : Set (FinsetAdeleRing R K S)} (h : U ∈ Filter.comap ι(S) (nhds (ι(S) x))) :
  U ∈ nhds x := by ...

theorem algebraMap_inducing : Inducing ι(S) := by
 refine inducing_iff_nhds.2
  (fun x => Filter.ext (fun U => ⟨fun hU => ⟨ι(S) '' U, ?_⟩,
   mem_nhds_comap_algebraMap x⟩))
 exact ⟨algebraMap_image_mem_nhds x hU,
  by rw [(algebraMap_injective R K S).preimage_image]⟩
```

Maps that are `Inducing` are also open and continuous, so we can use $\iota(S)$ to push forward compact neighbourhoods from the finite S-adele ring to the finite adele ring as in the informal proof of Theorem 5.9. The formal proof may then be given as follows.

```
instance FiniteAdeleRing.locallyCompactSpace :
  LocallyCompactSpace (FiniteAdeleRing R K) := by
 refine LocallyCompactSpace.mk <| fun x N hN => let S := support x; ?_
 have h := (algebraMap_inducing R K S).nhds_eq_comap (ofFiniteAdeleSupport x)
 let ⟨M, hM⟩ := (FinsetAdeleRing.locallyCompactSpace R K S).local_compact_nhds
  (ofFiniteAdeleSupport x) _ (h ▸ Filter.preimage_mem_comap hN)
 refine ⟨ι (S) '' M, ?_, Set.image_subset_iff.2 hM.2.1,
  (algebraMap_inducing R K S).isCompact_iff.1 hM.2.2⟩
 have h := algebraMap_range_mem_nhds (ofFiniteAdeleSupport x)
 exact (algebraMap_inducing R K S).map_nhds_of_mem _ h ▸ Filter.image_mem_map hM.1

end DedekindDomain
```

5.3 Local compactness of the adele ring

Theorem 5.11. *The adele ring $\mathbb{A}_K$ of a number field K is locally compact.*

Proof. As the product of the infinite adele ring and the finite adele ring, which are each locally compact (Theorems 5.3 and 5.9, respectively), the adele ring is also locally compact. □

```
instance NumberField.AdeleRing.locallyCompactSpace (K : Type*) [Field K]
  [NumberField K] : LocallyCompactSpace (AdeleRing K) :=
 Prod.locallyCompactSpace _ _
```

6 Discussion

6.1 A comparison of approaches for handling multiple instances

There are many cases in mathematics where we would like multiple distinct instances of the same class on a type. An example appeared in this research, where we had distinct uniform structures coming from the infinite places of a number field. There are a number of methods for assigning such instances in Lean, where the type class inference system expects only a single instance of a particular class to be assigned to a type. The approach we took to resolve this issue in Section 3.1 was to define the type synonym `WithAbs`. We will compare the type synonym approach with two other approaches in this section.

Throughout Section 6.1 the following variables are in scope.

```
variable {K : Type*} [Field K] (v : AbsoluteValue K ℝ)
```

6.1.1 Type synonyms

The basic usage of a type synonym is to create a copy of an already existing type. This enables one to assign an additional instance of a class to that type. The archetype of this in MATHLIB is `Lex`, which is defined simply as `Lex` $\alpha := \alpha$ and is used to give a type its lexicographic order.

It is possible to assign a family of instances by defining dependent type synonyms. This was the approach we took for `WithAbs`, which redefined a semiring as one that is dependent on absolute values. This approach creates separate indexed types, so that a single instance from the family is assigned to each, and type class search resolves automatically. In our application, we required `UniformSpace` instances coming from absolute values of a field so that we could apply `UniformSpace.Completion` as follows.

```
instance WithAbs.normedField : NormedField (WithAbs v) where ...

abbrev AbsoluteValue.Completion := UniformSpace.Completion (WithAbs v)
```

Notice that we do not need to tell the inference system which `UniformSpace` instance we really mean. In addition, type class search resolves automatically in dependent results. For example, we can state `CompletableTopField K` and apply `UniformSpace.Completion.instField` in the following without an explicit `UniformSpace` instance.

```
instance [CompletableTopField (WithAbs v)] : Field v.Completion :=
 UniformSpace.Completion.instField
```

6.1.2 Dependent constructors to a type class

Another approach to assigning multiple instances is to define an alternate constructor to the class we want to assign. In the case where we have an indexed family of instances, then the alternate constructor should depend on this index. A previous version of our work used this approach in order to specify instances coming from absolute values. In this case we define a constructor to the `NormedField` class from an absolute value.

```
instance AbsoluteValue.normedFieldCons : NormedField K where
 norm := v
 ...
```

The required `UniformSpace` instance on `K` associated to `v` can automatically be inferred from `v.normedFieldCons`. However, type class search cannot automatically find `v.normedFieldCons`, even when it is made an instance, due to the expectation that `K` have only a single `NormedField` instance. This approach has the drawback that we must provide the `v.normedFieldCons` instance whenever required, as follows.

```
abbrev AbsoluteValue.CompletionCons :=
 letI := v.normedFieldCons -- Explicit instance required
 UniformSpace.Completion K

instance [letI := v.normedFieldCons; CompletableTopField K] :
  Field v.CompletionCons :=
 letI := v.normedFieldCons -- Explicit instance required
 UniformSpace.Completion.instField
```

6.1.3 Data-carrying type class

A third approach to assigning an indexed family of instances is to define a new class altogether, which equips the type with a choice of index. This is similar to the type synonym approach but the data is stored within a class now, as opposed to a new type.

```
class WithAbsReal (R : Type*) [Semiring R] where
 v : AbsoluteValue R ℝ

instance [WithAbsReal K] : NormedField K := WithAbsReal.v.normedFieldCons
```

Then we make an explicit instance of this type class within the completion operation.

```
abbrev AbsoluteValue.CompletionClass :=
 letI := WithAbsReal.mk v -- Explicit instance required
 UniformSpace.Completion K
```

As in the dependent constructor approach, any instance of a parent class of `NormedField` needs to be explicitly provided whenever required.

```
instance [letI := WithAbsReal.mk v; CompletableTopField K] :
  Field v.CompletionClass :=
 letI := WithAbsReal.mk v -- Explicit instance required
 UniformSpace.Completion.instField
```

Out of the three approaches described, the type synonym approach optimises type class resolution. This helps to ensure robust further usage, development and maintenance.

6.2 Completing a number field at an infinite place through subfields

In Section 4.1.2, we described the formalisation of the completion of a number field at an infinite place. The approach there hinged on the properties of the uniform space given by σ-induced absolute values, where $\sigma : K \hookrightarrow \mathbb{C}$. In this section, we describe an alternate

approach where we inject K onto its image first, to obtain a `Subfield` $\mathbb{C}$ term, and then apply `UniformSpace.Completion` to the subfield term.

```
variable {K : Type*} [Field K] (v : InfinitePlace K)

def NumberField.InfinitePlace.subfield : Subfield ℂ where
 toSubring := v.embedding.range ...

def toSubfield : K →+* v.subfield where ...

def NumberField.InfinitePlace.CompletionSubfield :=
 UniformSpace.Completion v.subfield

instance : Field v.CompletionSubfield := inferInstance
```

By injecting K to its image, we remove the issues around multiple `UniformSpace` instances that we described in Section 6.1. As a result, the type class inference system is immediately able to infer necessary instances such as `UniformSpace` and `Field`.

The subfield completion defined by `v.CompletionSubfield` is isomorphic to `v.Completion` of Section 4.1.2 as uniform spaces by Theorem 2.11, because their constructions both define abstract completions of K. The canonical norm on `v.CompletionSubfield` is the *complex* absolute value, but any $x \in K$ first embeds into `v.subfield` via σ before being coerced to `v.CompletionSubfield`, so $x \in K$ still has the value $|\sigma(x)|_{\mathbb{C}}$.

```
instance : Coe K v.CompletionSubfield where
 coe := (UniformSpace.Completion.coe' v.subfield) ∘ v.toSubfield
```

However, this approach has a drawback. Any uniform completion $(Y, \iota : K \to Y)$ of K comes with the property that, for any uniformly continuous homomorphism $f : K \to Z$ from K to a complete separated uniform field Z, there exists a uniformly continuous homomorphism $\hat{f} : Y \to Z$ such that $\hat{f}(\iota(x)) = f(x)$ for all $x \in K$. This map is given by `UniformSpace.Completion.extensionHom` in MATHLIB. In this alternate construction, however, we can only extend functions from `v.subfield` to `v.CompletionSubfield`, and not those from K without further work. Generally, in bypassing the use of `UniformSpace.Completion` directly on K, we do not have immediate access to parts of the `UniformSpace.Completion` API that transfer properties and functions of K to its completion. This makes it more difficult to maintain a useful API for `v.CompletionSubfield`.

6.3 Formalising the infinite adele ring as a mixed space

An infinite place $v \in \Sigma_{K,\infty}$ is *real* if the image of its associated embedding $K \hookrightarrow \mathbb{C}$ lies entirely within $\mathbb{R}$, in which case $K_v = \mathbb{R}$, otherwise it is *complex* and $K_v = \mathbb{C}$. Therefore the

infinite adele ring can also be viewed as $\mathbb{R}^{r_1} \times \mathbb{C}^{r_2}$, where r_1 and r_2 are the number of real and complex places respectively. This is the space with local notation `E K` in MATHLIB (redefined as `NumberField.mixedEmbedding.mixedSpace` in later versions).

We chose not to use `mixedSpace` as the formalisation of the infinite adele ring for a number of reasons. For one, it would mean the adele ring would be defined as a triple product over the two subtypes of `InfinitePlace` and the type of prime ideals on $\mathcal{O}_K$, which is more cumbersome. One could also define the completion of `K` at `v : InfinitePlace K` as `if v.IsReal then ℝ else ℂ`, but this is neither definitionally equal to $\mathbb{R}$ nor $\mathbb{C}$. Secondly, as discussed in Section 6.2, the `UniformSpace.Completion` API comes with useful constructions such as extensions of homomorphisms that allow us to transfer certain functions on K to its completion. We would not have immediate access to these with the `mixedSpace` formalisation.

In our work, we show isometric ring isomorphisms between K_v and $\mathbb{R}$ or $\mathbb{C}$ as appropriate, as well as an isomorphism between the infinite adele ring and the mixed space.

```
def InfiniteAdeleRing.ringEquivMixedSpace (K : Type*) [Field K] :
  InfiniteAdeleRing K ≃+* ({w : InfinitePlace K // IsReal w} → ℝ) ×
    ({w : InfinitePlace K // IsComplex w} → ℂ) := ...
```

6.4 Future work

This work establishes a key result in the early stages of the five-year effort to formalise Fermat's Last Theorem. It also represents a step towards the formalisation of Tate's thesis in MATHLIB, where the adele ring was used to establish the foundations of the Langlands program for $GL(1)$. Other theoretical ingredients for Tate's thesis, such as Haar measures [Doo21], are now also available in MATHLIB. We list some ordered future work to follow directly on from this research.

- Show that the adele ring of a function field is locally compact.
- Show that K is a discrete cocompact subgroup of $\mathbb{A}_K$.
- Prove that the idele group $\mathbb{I}_K$ is locally compact and $K^\times$ is a discrete cocompact subgroup of $\mathbb{I}_K$.
- Formalise adelic Hecke characters.
- Formalise Tate's thesis.

Acknowledgments

I would like to thank María Inés de Frutos-Fernández for their advice on formalising the infinite adele ring of a number field, and the MATHLIB community for their constructive

feedback on the code. I would also like to thank the anonymous reviewers for their helpful suggestions that have improved this paper.

References

[Bou66] N. Bourbaki. *Elements of mathematics. General topology. Part 1*. Hermann Paris; Addison-Wesley Publishing Co. Reading Mass.-London-Don Mills Ont., 1966, pages vii+437.

[BCM20] K. Buzzard, J. Commelin, and P. Massot. "Formalising perfectoid spaces". In: *Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs*. CPP 2020. New Orleans, LA, USA: Association for Computing Machinery, 2020, pages 299–312. ISBN: 9781450370974. DOI: 10.1145/3372885.3373830. URL: https://doi.org/10.1145/3372885.3373830.

[CF67] J. W. S. Cassels and A. Fröhlich. *Algebraic number theory*. Academic Press London; Thompson Book Co. Inc., Washington D.C., 1967, pages xvii + 366.

[Doo21] F. van Doorn. "Formalized Haar Measure". In: *12th International Conference on Interactive Theorem Proving (ITP 2021)*. Edited by L. Cohen and C. Kaliszyk. Volume 193. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021, 18:1–18:17. ISBN: 978-3-95977-188-7. DOI: 10.4230/LIPIcs.ITP.2021.18. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.ITP.2021.18.

[FF22] M. I. de Frutos-Fernández. "Formalizing the Ring of Adèles of a Global Field". In: *13th International Conference on Interactive Theorem Proving (ITP 2022)*. Edited by J. Andronick and L. de Moura. Volume 237. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022, 14:1–14:18. ISBN: 978-3-95977-252-5. DOI: 10.4230/LIPIcs.ITP.2022.14. URL: https://drops.dagstuhl.de/opus/volltexte/2022/16723.

[FFN24] M. I. de Frutos-Fernández and F. A. E. Nuccio Mortarino Majno di Capriglio. "A Formalization of Complete Discrete Valuation Rings and Local Fields". In: *Proceedings of the 13th ACM SIGPLAN International Conference on Certified Programs and Proofs*. CPP 2024. New York, NY, USA: Association for Computing Machinery, 2024, pages 190–204. ISBN: 9798400704888. DOI: 10.1145/3636501.3636942. URL: https://doi.org/10.1145/3636501.3636942.

[mat20] The mathlib Community. “The Lean Mathematical Library”. In: *Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs*. CPP 2020. New Orleans, LA, USA: Association for Computing Machinery, Jan. 2020, 367–381. DOI: 10.1145/3372885.3373824. URL: https://doi.org/10.1145/3372885.3373824.

[Ost16] A. Ostrowski. “Über einige Lösungen der Funktionalgleichung $\psi(x) \cdot \psi(x) = \psi(xy)$”. In: *Acta Mathematica* 41.none (1916), pages 271 –284. DOI: 10.1007/BF02422947. URL: https://doi.org/10.1007/BF02422947.

[Ser79] J.-P. Serre. *Local Fields*. Graduate Texts in Mathematics. New York: Springer New York, 1979. ISBN: 9780387904245. DOI: https://doi.org/10.1007/978-1-4757-5673-9.

[Tat50] J. T. Tate. “Fourier analysis in number fields, and Hecke’s zeta-functions”. In: *Algebraic Number Theory*. Proc. Instructional Conf., Brighton, 1965. 1950. URL: https://api.semanticscholar.org/CorpusID:117898073.

[Wei38] A. Weil. *Sur les espaces à structure uniforme et sur la topologie générale*. Actualités scientifiques et industrielles. Hermann & cie, 1938. URL: https://books.google.co.uk/books?id=qVfvAAAAMAAJ.