

Furstenberg's Topology And His Proof of the Infinitude of Primes

Manuel Eberl

February 6, 2026

Abstract

This article gives a formal version of Furstenberg's topological proof of the infinitude of primes. He defines a topology on the integers based on arithmetic progressions (or, equivalently, residue classes). Using some fairly obvious properties of this topology, the infinitude of primes is then easily obtained.

Apart from this, this topology is also fairly 'nice' in general: it is second countable, metrizable, and perfect. All of these (well-known) facts are formally proven, including an explicit metric for the topology given by Zulfeqarr.

Contents

1 Furstenberg's topology and his proof of the infinitude of primes	2
1.1 Arithmetic progressions of integers	2
1.2 The Furstenberg topology on \mathbb{Z}	3
1.3 The infinitude of primes	6
1.4 Additional topological properties	7
1.5 Metrizability	10

1 Furstenberg's topology and his proof of the infinitude of primes

```
theory Furstenberg-Topology
imports
  HOL-Real-Asymp.Real-Asymp
  HOL-Analysis.Analysis
  HOL-Number-Theory.Number-Theory
begin
```

This article gives a formal version of Furstenberg's topological proof of the infinitude of primes [2]. He defines a topology on the integers based on arithmetic progressions (or, equivalently, residue classes).

Apart from yielding a short proof of the infinitude of primes, this topology is also fairly 'nice' in general: it is second countable, metrizable, and perfect. All of these (well-known) facts will be formally proven below.

1.1 Arithmetic progressions of integers

We first define 'bidirectional infinite arithmetic progressions' on \mathbb{Z} in the sense that to an integer a and a positive integer b , we associate all the integers x such that $x \equiv a \pmod{b}$, or, equivalently, $\{a + nb \mid n \in \mathbb{Z}\}$.

```
definition arith-prog :: int  $\Rightarrow$  nat  $\Rightarrow$  int set where
  arith-prog a b =  $\{x. [x = a] \pmod{int\ b}\}$ 
```

```
lemma arith-prog-0-right [simp]: arith-prog a 0 =  $\{a\}$ 
by (simp add: arith-prog-def)
```

```
lemma arith-prog-Suc-0-right [simp]: arith-prog a (Suc 0) = UNIV
by (auto simp: arith-prog-def)
```

```
lemma in-arith-progI [intro]:  $[x = a] \pmod{b} \implies x \in \textit{arith-prog } a\ b$ 
by (auto simp: arith-prog-def)
```

Two arithmetic progressions with the same period and noncongruent starting points are disjoint.

```
lemma arith-prog-disjoint:
  assumes  $[a \neq a'] \pmod{int\ b}$  and  $b > 0$ 
  shows  $\textit{arith-prog } a\ b \cap \textit{arith-prog } a'\ b = \{\}$ 
  using assms by (auto simp: arith-prog-def cong-def)
```

Multiplying the period gives us a subset of the original progression.

```
lemma arith-prog-dvd-mono:  $b\ \textit{dvd}\ b' \implies \textit{arith-prog } a\ b' \subseteq \textit{arith-prog } a\ b$ 
by (auto simp: arith-prog-def cong-dvd-modulus)
```

The following proves the alternative definition mentioned above.

```

lemma bij-betw-arith-prog:
  assumes  $b > 0$ 
  shows  $\text{bij-betw } (\lambda n. a + \text{int } b * n) \text{ UNIV } (\text{arith-prog } a \ b)$ 
proof (rule bij-betwI[of - - -  $\lambda x. (x - a) \text{ div int } b$ ], goal-cases)
  case 1
  thus ?case
  by (auto simp: arith-prog-def cong-add-lcancel-0 cong-mult-self-right mult-of-nat-commute)
next
  case 4
  thus ?case
  by (auto simp: arith-prog-def cong-iff-lin)
qed (use  $\langle b > 0 \rangle$  in  $\langle \text{auto simp: arith-prog-def} \rangle$ )

```

```

lemma arith-prog-altdef:  $\text{arith-prog } a \ b = \text{range } (\lambda n. a + \text{int } b * n)$ 
proof (cases  $b = 0$ )
  case False
  thus ?thesis
  using bij-betw-arith-prog[of  $b$ ] by (auto simp: bij-betw-def)
qed auto

```

A simple corollary from this is also that any such arithmetic progression is infinite.

```

lemma infinite-arith-prog:  $b > 0 \implies \text{infinite } (\text{arith-prog } a \ b)$ 
  using bij-betw-finite[OF bij-betw-arith-prog[of  $b$ ]] by simp

```

1.2 The Furstenberg topology on \mathbb{Z}

The typeclass-based topology is somewhat nicer to use in Isabelle/HOL, but the integers, of course, already have a topology associated to them. We therefore need to introduce a type copy of the integers and furnish them with the new topology. We can easily convert between them and the ‘proper’ integers using Lifting and Transfer.

```

typedef fbint = UNIV :: int set
  morphisms int-of-fbint fbint ..

```

```

setup-lifting type-definition-fbint

```

```

lift-definition arith-prog-fb ::  $\text{int} \Rightarrow \text{nat} \Rightarrow \text{fbint set}$  is arith-prog .

```

```

instantiation fbint :: topological-space
begin

```

Furstenberg defined the topology as the one generated by all arithmetic progressions. We use a slightly more explicit equivalent formulation that exploits the fact that the intersection of two arithmetic progressions is again an arithmetic progression (or empty).

```

lift-definition open-fbint ::  $\text{fbint set} \Rightarrow \text{bool}$  is

```

$\lambda U. (\forall x \in U. \exists b > 0. \text{arith-prog } x \ b \subseteq U) .$

We now prove that this indeed forms a topology.

instance proof

show *open* (*UNIV* :: *fbint set*)
by *transfer auto*

next

fix *U V* :: *fbint set*

assume *open U* **and** *open V*

show *open (U ∩ V)*

proof (*use* $\langle \text{open } U \rangle \langle \text{open } V \rangle$ **in** *transfer, safe*)

fix *U V* :: *int set* **and** *x* :: *int*

assume $U: \forall x \in U. \exists b > 0. \text{arith-prog } x \ b \subseteq U$ **and** $V: \forall x \in V. \exists b > 0. \text{arith-prog } x \ b \subseteq V$

assume $x: x \in U \ x \in V$

from *U x* **obtain** *b1* **where** $b1: b1 > 0 \ \text{arith-prog } x \ b1 \subseteq U$ **by** *auto*

from *V x* **obtain** *b2* **where** $b2: b2 > 0 \ \text{arith-prog } x \ b2 \subseteq V$ **by** *auto*

from *b1 b2* **have** $\text{lcm } b1 \ b2 > 0 \ \text{arith-prog } x \ (\text{lcm } b1 \ b2) \subseteq U \cap V$

using *arith-prog-dvd-mono*[*of b1 lcm b1 b2 x*] *arith-prog-dvd-mono*[*of b2 lcm b1 b2 x*]

by (*auto simp: lcm-pos-nat*)

thus $\exists b > 0. \text{arith-prog } x \ b \subseteq U \cap V$ **by** *blast*

qed

next

fix *F* :: *fbint set set*

assume $*$: $\forall U \in F. \text{open } U$

show *open* ($\bigcup F$)

proof (*use* $*$ **in** *transfer, safe*)

fix *F* :: *int set set* **and** *U* :: *int set* **and** *x* :: *int*

assume $F: \forall U \in F. \forall x \in U. \exists b > 0. \text{arith-prog } x \ b \subseteq U$

assume $x \in U \ U \in F$

with *F* **obtain** *b* **where** $b: b > 0 \ \text{arith-prog } x \ b \subseteq U$ **by** *blast*

with $\langle U \in F \rangle$ **show** $\exists b > 0. \text{arith-prog } x \ b \subseteq \bigcup F$

by *blast*

qed

qed

end

Since any non-empty open set contains an arithmetic progression and arithmetic progressions are infinite, we obtain that all nonempty open sets are infinite.

lemma *open-fbint-imp-infinite*:

fixes *U* :: *fbint set*

assumes *open U* **and** $U \neq \{\}$

shows *infinite U*

using *assms*

proof *transfer*

fix *U* :: *int set*

```

assume *:  $\forall x \in U. \exists b > 0. \text{arith-prog } x \ b \subseteq U$  and  $U \neq \{\}$ 
from  $\langle U \neq \{\} \rangle$  obtain  $x$  where  $x \in U$  by auto
with * obtain  $b$  where  $b > 0$  arith-prog  $x \ b \subseteq U$  by auto
from  $b$  have infinite (arith-prog  $x \ b$ )
  using infinite-arith-prog by blast
with  $b$  show infinite  $U$ 
  using finite-subset by blast
qed

```

```

lemma not-open-finite-fbint [simp]:
  assumes finite ( $U :: \text{fbint set}$ )  $U \neq \{\}$ 
  shows  $\neg \text{open } U$ 
  using open-fbint-imp-infinite assms by blast

```

More or less by definition, any arithmetic progression is open.

```

lemma open-arith-prog-fb [intro]:
  assumes  $b > 0$ 
  shows open (arith-prog-fb  $a \ b$ )
  using assms
proof transfer
  fix  $a :: \text{int}$  and  $b :: \text{nat}$ 
  assume  $b > 0$ 
  show  $\forall x \in \text{arith-prog } a \ b. \exists b' > 0. \text{arith-prog } x \ b' \subseteq \text{arith-prog } a \ b$ 
  proof (intro ball exI[of -] conjI)
    fix  $x$  assume  $x \in \text{arith-prog } a \ b$ 
    thus arith-prog  $x \ b \subseteq \text{arith-prog } a \ b$ 
    using cong-trans by (auto simp: arith-prog-def)
  qed (use  $\langle b > 0 \rangle$  in auto)
qed

```

Slightly less obviously, any arithmetic progression is also closed. This can be seen by realising that for a period b , we can partition the integers into b congruence classes and then the complement of each congruence class is the union of the other $b - 1$ classes, and unions of open sets are open.

```

lemma closed-arith-prog-fb [intro]:
  assumes  $b > 0$ 
  shows closed (arith-prog-fb  $a \ b$ )
proof -
  have open ( $\neg \text{arith-prog-fb } a \ b$ )
  proof -
    have  $\neg \text{arith-prog-fb } a \ b = (\bigcup_{i \in \{1..<b\}}. \text{arith-prog-fb } (a+i) \ b)$ 
    proof (transfer fixing: b)
      fix  $a :: \text{int}$ 
      have disjoint:  $x \notin \text{arith-prog } a \ b$  if  $x \in \text{arith-prog } (a + \text{int } i) \ b \ i \in \{1..<b\}$ 
for  $x \ i$ 
      proof -
        have  $[a \neq a + \text{int } i] \ (\text{mod } \text{int } b)$ 
        proof
          assume  $[a = a + \text{int } i] \ (\text{mod } \text{int } b)$ 

```

```

    hence [a + 0 = a + int i] (mod int b) by simp
    hence [0 = int i] (mod int b) by (subst (asm) cong-add-lcancel) auto
    with that show False by (auto simp: cong-def)
  qed
  thus ?thesis using arith-prog-disjoint[of a a + int i b] ⟨b > 0⟩ that by auto
  qed

  have covering: x ∈ arith-prog a b ∨ x ∈ (⋃ i∈{1..<b}. arith-prog (a + int i)
  b) for x
  proof -
    define i where i = nat ((x - a) mod (int b))
    have [a + int i = a + (x - a) mod int b] (mod int b)
      unfolding i-def using ⟨b > 0⟩ by simp
    also have [a + (x - a) mod int b = a + (x - a)] (mod int b)
      by (intro cong-add) auto
    finally have [x = a + int i] (mod int b)
      by (simp add: cong-sym-eq)
    hence x ∈ arith-prog (a + int i) b
      using ⟨b > 0⟩ by (auto simp: arith-prog-def)
    moreover have i < b using ⟨b > 0⟩
      by (auto simp: i-def nat-less-iff)
    ultimately show ?thesis using ⟨b > 0⟩
      by (cases i = 0) auto
  qed

  from disjoint and covering show - arith-prog a b = (⋃ i∈{1..<b}. arith-prog
  (a + int i) b)
  by blast
  qed
  also from ⟨b > 0⟩ have open ...
  by auto
  finally show ?thesis .
  qed
  thus ?thesis by (simp add: closed-def)
  qed

```

1.3 The infinitude of primes

The infinite of the primes now follows quite obviously: The multiples of any prime form a closed set, so if there were only finitely many primes, the union of all of these would also be open. However, since any number other than ± 1 has a prime divisor, the union of all these sets is simply $\mathbb{Z} \setminus \{\pm 1\}$, which is obviously *not* closed since the finite set $\{\pm 1\}$ is not open.

```

theorem infinite {p::nat. prime p}
proof
  assume fin: finite {p::nat. prime p}
  define A where A = (⋃ p∈{p::nat. prime p}. arith-prog-fb 0 p)
  have closed A

```

```

  unfolding A-def using fin by (intro closed-Union) (auto simp: prime-gt-0-nat)
  hence open (-A)
  by (simp add: closed-def)
  also have A = -{fbint 1, fbint (-1)}
  unfolding A-def
  proof transfer
  show (⋃ p∈{p::nat. prime p}. arith-prog 0 p) = - {1, - 1}
  proof (intro equalityI subsetI)
  fix x :: int assume x: x ∈ -{1, -1}
  hence |x| ≠ 1 by auto
  show x ∈ (⋃ p∈{p::nat. prime p}. arith-prog 0 p)
  proof (cases x = 0)
  case True
  thus ?thesis
  by (auto simp: A-def intro!: exI[of - 2])
  next
  case [simp]: False
  obtain p where p: prime p p dvd x
  using prime-divisor-exists[of x] and ‹|x| ≠ 1› by auto
  hence x ∈ arith-prog 0 (nat p) using prime-gt-0-int[of p]
  by (auto simp: arith-prog-def cong-0-iff)
  thus ?thesis using p
  by (auto simp: A-def intro!: exI[of - nat p])
  qed
  qed (auto simp: A-def arith-prog-def cong-0-iff)
  qed
  also have -(-{fbint 1, fbint (-1)}) = {fbint 1, fbint (-1)}
  by simp
  finally have open {fbint 1, fbint (-1)} .
  thus False by simp
  qed

```

1.4 Additional topological properties

Just for fun, let us also show a few more properties of Furstenberg's topology. First, we show the equivalence to the above to Furstenberg's original definition (the topology generated by all arithmetic progressions).

theorem *topological-basis-fbint*: *topological-basis* {arith-prog-fb a b | a b. b > 0}

unfolding *topological-basis-def*

proof *safe*

fix a :: int **and** b :: nat

assume b > 0

thus open (arith-prog-fb a b)

by *auto*

next

fix U :: fbint set **assume** open U

hence $\forall x \in U. \exists b. b > 0 \wedge \text{arith-prog-fb } (int\text{-of-fbint } x) b \subseteq U$

by *transfer*

hence $\exists f. \forall x \in U. f x > 0 \wedge \text{arith-prog-fb } (int\text{-of-fbint } x) (f x) \subseteq U$

```

    by (subst (asm) bchoice-iff)
  then obtain f where f:  $\forall x \in U. f x > 0 \wedge \text{arith-prog-fb } (\text{int-of-fbint } x) (f x) \subseteq U$  ..
  define B where B =  $(\lambda x. \text{arith-prog-fb } (\text{int-of-fbint } x) (f x)) \text{ ' } U$ 
  have B  $\subseteq \{\text{arith-prog-fb } a b \mid a b. b > 0\}$ 
    using f by (auto simp: B-def)
  moreover have  $\bigcup B = U$ 
  proof safe
    fix x assume x  $\in U$ 
    hence x  $\in \text{arith-prog-fb } (\text{int-of-fbint } x) (f x)$ 
      using f by transfer auto
    with  $\langle x \in U \rangle$  show x  $\in \bigcup B$  by (auto simp: B-def)
  qed (use f in <auto simp: B-def>)
  ultimately show  $\exists B' \subseteq \{\text{arith-prog-fb } a b \mid a b. 0 < b\}. \bigcup B' = U$  by auto
  qed

```

```

lemma open-fbint-altdef: open = generate-topology  $\{\text{arith-prog-fb } a b \mid a b. b > 0\}$ 
  using topological-basis-imp-subbasis[OF topological-basis-fbint] .

```

From this, we can immediately see that it is second countable:

```

instance fbint :: second-countable-topology
proof
  have countable  $((\lambda(a,b). \text{arith-prog-fb } a b) \text{ ' } (UNIV \times \{b. b > 0\}))$ 
    by (intro countable-image) auto
  also have  $\dots = \{\text{arith-prog-fb } a b \mid a b. b > 0\}$ 
    by auto
  ultimately show  $\exists B::\text{fbint set set. countable } B \wedge \text{open} = \text{generate-topology } B$ 
    unfolding open-fbint-altdef by auto
  qed

```

A trivial consequence of the fact that nonempty open sets in this topology are infinite is that it is a perfect space:

```

instance fbint :: perfect-space
  by standard auto

```

It is also Hausdorff, since given any two distinct integers, we can easily construct two non-overlapping arithmetic progressions that each contain one of them. We do not *really* have to prove this since we will get it for free later on when we show that it is a metric space, but here is the proof anyway:

```

instance fbint :: t2-space
proof
  fix x y :: fbint
  assume x  $\neq$  y
  define d where d = nat  $|\text{int-of-fbint } x - \text{int-of-fbint } y| + 1$ 
  from  $\langle x \neq y \rangle$  have d  $> 0$ 
    unfolding d-def by transfer auto
  define U where U =  $\text{arith-prog-fb } (\text{int-of-fbint } x) d$ 
  define V where V =  $\text{arith-prog-fb } (\text{int-of-fbint } y) d$ 

```

```

have  $U \cap V = \{\}$  unfolding U-def V-def d-def
proof (use  $\langle x \neq y \rangle$  in transfer, rule arith-prog-disjoint)
  fix  $x y :: \text{int}$ 
  assume  $x \neq y$ 
  show  $[x \neq y] \pmod{\text{int}} (\text{nat } |x - y| + 1)$ 
  proof
    assume  $[x = y] \pmod{\text{int}} (\text{nat } |x - y| + 1)$ 
    hence  $|x - y| + 1 \text{ dvd } |x - y|$ 
    by (auto simp: cong-iff-dvd-diff algebra-simps)
    hence  $|x - y| + 1 \leq |x - y|$ 
    by (rule zdvd-imp-le) (use  $\langle x \neq y \rangle$  in auto)
  thus False by simp
  qed
qed auto
moreover have  $x \in U \wedge y \in V$ 
  unfolding U-def V-def by (use  $\langle d > 0 \rangle$  in transfer, fastforce)+
moreover have open U open V
  using  $\langle d > 0 \rangle$  by (auto simp: U-def V-def)
ultimately show  $\exists U V. \text{open } U \wedge \text{open } V \wedge x \in U \wedge y \in V \wedge U \cap V = \{\}$ 
by blast
qed

```

Next, we need a small lemma: Given an additional assumption, a T_2 space is also T_3 :

```

lemma t2-space-t3-spaceI:
  assumes  $\bigwedge (x :: 'a :: \text{t2-space}) U. x \in U \implies \text{open } U \implies$ 
     $\exists V. x \in V \wedge \text{open } V \wedge \text{closure } V \subseteq U$ 
  shows OFCLASS('a, t3-space-class)
proof
  fix  $X :: 'a \text{ set}$  and  $z :: 'a$ 
  assume  $X: \text{closed } X \wedge z \notin X$ 
  with assms[of z - X] obtain  $V$  where  $z \in V \wedge \text{open } V \wedge \text{closure } V \subseteq -X$ 
  by auto
  show  $\exists U V. \text{open } U \wedge \text{open } V \wedge z \in U \wedge X \subseteq V \wedge U \cap V = \{\}$ 
  by (rule exI[of - V], rule exI[of - -closure V])
    (use  $X \text{ closure-subset}[of V]$  in auto)
qed

```

Since the Furstenberg topology is T_2 and every arithmetic progression is also closed, we can now easily show that it is also T_3 (i. e. regular). Again, we do not really need this proof, but here it is:

```

instance fbint :: t3-space
proof (rule t2-space-t3-spaceI)
  fix  $x :: \text{fbint}$  and  $U :: \text{fbint set}$ 
  assume  $x \in U$  and open U
  then obtain  $b$  where  $b > 0$  arith-prog-fb (int-of-fbint x) b  $b \subseteq U$ 
  by transfer blast
  define  $V$  where  $V = \text{arith-prog-fb (int-of-fbint x) b}$ 

```

```

have  $x \in V$ 
  unfolding  $V$ -def by transfer auto
moreover have open  $V$  closed  $V$ 
  using  $\langle b > 0 \rangle$  by (auto simp:  $V$ -def)
ultimately show  $\exists V. x \in V \wedge \text{open } V \wedge \text{closure } V \subseteq U$ 
  using  $b$  by (intro exI[of -  $V$ ]) (auto simp:  $V$ -def)
qed

```

1.5 Metrizable

The metrizable of Furstenberg's topology (i. e. that it is induced by some metric) can be shown from the fact that it is second countable and T_3 using Urysohn's Metrization Theorem, but this is not available in Isabelle yet. Let us therefore give an *explicit* metric, as described by Zulfeqarr [3]. We follow the exposition by Dirmeier [1].

First, we define a kind of norm on the integers. The norm depends on a real parameter $q > 1$. The value of q does not matter in the sense that all values induce the same topology (which we will show). For the final definition, we then simply pick $q = 2$.

```

locale  $fbnorm =$ 
  fixes  $q :: \text{real}$ 
  assumes  $q\text{-gt-1}: q > 1$ 
begin

```

definition $N :: \text{int} \Rightarrow \text{real}$ **where**

$$N\ n = (\sum k. \text{if } k = 0 \vee \text{int } k \text{ dvd } n \text{ then } 0 \text{ else } 1 / q^{\wedge} k)$$

lemma N -summable: summable $(\lambda k. \text{if } k = 0 \vee \text{int } k \text{ dvd } n \text{ then } 0 \text{ else } 1 / q^{\wedge} k)$

```

by (rule summable-comparison-test[OF - summable-geometric[of 1/q]])
  (use  $q\text{-gt-1}$  in  $\langle$ auto intro!: exI[of - 0] simp: power-divide $\rangle$ )

```

lemma N -sums: $(\lambda k. \text{if } k = 0 \vee \text{int } k \text{ dvd } n \text{ then } 0 \text{ else } 1 / q^{\wedge} k)$ sums $N\ n$

```

using  $N$ -summable unfolding  $N$ -def by (rule summable-sums)

```

lemma N -nonneg: $N\ n \geq 0$

```

by (rule sums-le[OF - sums-zero  $N$ -sums]) (use  $q\text{-gt-1}$  in auto)

```

lemma N -uminus [simp]: $N\ (-n) = N\ n$

```

by (simp add:  $N$ -def)

```

lemma N -minus-commute: $N\ (x - y) = N\ (y - x)$

```

using  $N$ -uminus[of  $x - y$ ] by (simp del:  $N$ -uminus)

```

lemma N -zero [simp]: $N\ 0 = 0$

```

by (simp add:  $N$ -def)

```

lemma not-dvd-imp- N -ge:

```

assumes  $\neg n \text{ dvd } a$   $n > 0$ 

```

shows $N a \geq 1 / q \wedge n$
by (*rule sums-le[OF - sums-single[of n] N-sums]*) (*use q-gt-1 assms in auto*)

lemma *N-lt-imp-dvd*:
assumes $N a < 1 / q \wedge n$ **and** $n > 0$
shows $n \text{ dvd } a$
using *not-dvd-imp-N-ge[of n a] assms* **by** *auto*

lemma *N-pos*:
assumes $n \neq 0$
shows $N n > 0$
proof –
have $0 < 1 / q \wedge (\text{nat } |n| + 1)$
using *q-gt-1* **by** *simp*
also have $\neg 1 + |n| \text{ dvd } |n|$
using *zdvd-imp-le[of 1 + |n| |n|] assms* **by** *auto*
hence $1 / q \wedge (\text{nat } |n| + 1) \leq N n$
by (*intro not-dvd-imp-N-ge*) (*use assms in auto*)
finally show *?thesis* .
qed

lemma *N-zero-iff [simp]*: $N n = 0 \longleftrightarrow n = 0$
using *N-pos[of n]* **by** (*cases n = 0*) *auto*

lemma *N-triangle-ineq*: $N (n + m) \leq N n + N m$
proof (*rule sums-le*)
let $?I = \lambda n k. \text{if } k = 0 \vee \text{int } k \text{ dvd } n \text{ then } 0 \text{ else } 1 / q \wedge k$
show $?I (n + m) \text{ sums } N (n + m)$
by (*rule N-sums*)
show $(\lambda k. ?I n k + ?I m k) \text{ sums } (N n + N m)$
by (*intro sums-add N-sums*)
qed (*use q-gt-1 in auto*)

lemma *N-1*: $N 1 = 1 / (q * (q - 1))$
proof (*rule sums-unique2*)
have $(\lambda k. \text{if } k = 0 \vee \text{int } k \text{ dvd } 1 \text{ then } 0 \text{ else } 1 / q \wedge k) \text{ sums } N 1$
by (*rule N-sums*)
also have $(\lambda k. \text{if } k = 0 \vee \text{int } k \text{ dvd } 1 \text{ then } 0 \text{ else } 1 / q \wedge k) =$
 $(\lambda k. \text{if } k \in \{0, 1\} \text{ then } 0 \text{ else } (1 / q) \wedge k)$
by (*simp add: power-divide cong: if-cong*)
finally show $(\lambda k. \text{if } k \in \{0, 1\} \text{ then } 0 \text{ else } (1 / q) \wedge k) \text{ sums } N 1$.

have $(\lambda k. \text{if } k \in \{0, 1\} \text{ then } 0 \text{ else } (1 / q) \wedge k) \text{ sums}$
 $(1 / (1 - 1 / q) + (- (1 / q) - 1))$
by (*rule sums-If-finite-set'[OF geometric-sums]*) (*use q-gt-1 in auto*)
also have $\dots = 1 / (q * (q - 1))$
using *q-gt-1* **by** (*simp add: field-simps*)
finally show $(\lambda k. \text{if } k \in \{0, 1\} \text{ then } 0 \text{ else } (1 / q) \wedge k) \text{ sums } \dots$.
qed

It follows directly from the definition that norms fulfil a kind of monotonicity property with respect to divisibility: the norm of a number is at most as large as the norm of any of its factors:

```

lemma N-dvd-mono:
  assumes m dvd n
  shows  $N\ n \leq N\ m$ 
proof (rule sums-le[OF - N-sums N-sums])
  fix k :: nat
  show (if k = 0  $\vee$  int k dvd n then 0 else  $1 / q^{\wedge} k$ )  $\leq$ 
    (if k = 0  $\vee$  int k dvd m then 0 else  $1 / q^{\wedge} k$ )
  using q-gt-1 assms by auto
qed

```

In particular, this means that 1 and -1 have the greatest norm.

```

lemma N-le-N-1:  $N\ n \leq N\ 1$ 
  by (rule N-dvd-mono) auto

```

Primes have relatively large norms, almost reaching the norm of 1:

```

lemma N-prime:
  assumes prime p
  shows  $N\ p = N\ 1 - 1 / q^{\wedge} \text{nat } p$ 
proof (rule sums-unique2)
  define p' where p' = nat p
  have p: p = int p'
    using assms by (auto simp: p'-def prime-ge-0-int)
  have prime p'
    using assms by (simp add: p)

  have ( $\lambda k. \text{if } k = 0 \vee \text{int } k \text{ dvd } p \text{ then } 0 \text{ else } 1 / q^{\wedge} k$ ) sums  $N\ p$ 
    by (rule N-sums)
  also have int k dvd p  $\longleftrightarrow k \in \{1, p'\}$  for k
    using assms by (auto simp: p prime-nat-iff)
  hence ( $\lambda k. \text{if } k = 0 \vee \text{int } k \text{ dvd } p \text{ then } 0 \text{ else } 1 / q^{\wedge} k$ ) =
    ( $\lambda k. \text{if } k \in \{0, 1, p'\} \text{ then } 0 \text{ else } (1 / q)^{\wedge} k$ )
    using assms q-gt-1 by (simp add: power-divide cong: if-cong)
  finally show ... sums  $N\ p$  .

  have ( $\lambda k. \text{if } k \in \{0, 1, p'\} \text{ then } 0 \text{ else } (1 / q)^{\wedge} k$ ) sums
    ( $1 / (1 - 1 / q) + (- (1 / q) - (1 / q)^{\wedge} p' - 1)$ )
    by (rule sums-If-finite-set'[OF geometric-sums])
    (use  $\langle \text{prime } p' \rangle$  q-gt-1 prime-gt-Suc-0-nat[of p'] in  $\langle \text{auto simp: } \rangle$ )
  also have ... =  $N\ 1 - 1 / q^{\wedge} p'$ 
    using q-gt-1 by (simp add: field-simps N-1)
  finally show ( $\lambda k. \text{if } k \in \{0, 1, p'\} \text{ then } 0 \text{ else } (1 / q)^{\wedge} k$ ) sums ... .
qed

```

```

lemma N-2:  $N\ 2 = 1 / (q^{\wedge} 2 * (q - 1))$ 
  using q-gt-1 by (auto simp: N-prime N-1 field-simps power2-eq-square)

```

lemma *N-less-N-1*:
assumes $n \neq 1$ $n \neq -1$
shows $N\ n < N\ 1$
proof (*cases* $n = 0$)
case *False*
then obtain p **where** p : *prime* $p\ p\ \text{dvd}\ n$
using *prime-divisor-exists*[*of* n] *assms* **by** *force*
hence $N\ n \leq N\ p$ **by** (*intro* *N-dvd-mono*)
also from p **have** $N\ p < N\ 1$
using *q-gt-1* **by** (*simp* *add: N-prime*)
finally show *?thesis* .
qed (*use* *q-gt-1* **in** \langle *auto* *simp: N-1* \rangle)

Composites, on the other hand, do not achieve this:

lemma *nonprime-imp-N-lt*:
assumes \neg *prime-elem* $n\ |n| \neq 1\ n \neq 0$
shows $N\ n < N\ 1 - 1 / q \wedge \text{nat}\ |n|$
proof –
obtain p **where** p : *prime* $p\ p\ \text{dvd}\ n$
using *prime-divisor-exists*[*of* n] *assms* **by** *auto*
define p' **where** $p' = \text{nat}\ p$
have p' : $p = \text{int}\ p'$
using p **by** (*auto* *simp: p'-def* *prime-ge-0-int*)
have *prime* p'
using p **by** (*simp* *add: p'*)

define n' **where** $n' = \text{nat}\ |n|$
have $n' > 1$
using *assms* **by** (*auto* *simp: n'-def*)

have $N\ n \leq 1 / (q * (q - 1)) - 1 / q \wedge p' - 1 / q \wedge n'$
proof (*rule* *sums-le*)
show $(\lambda k. \text{if } k = 0 \vee \text{int } k\ \text{dvd}\ n \text{ then } 0 \text{ else } 1 / q \wedge k)$ *sums* $N\ n$
by (*rule* *N-sums*)
next
from *assms* p **have** $n' \neq p'$
by (*auto* *simp: n'-def p'-def* *nat-eq-iff*)
hence $(\lambda k. \text{if } k \in \{0, 1, p', n'\} \text{ then } 0 \text{ else } (1 / q) \wedge k)$ *sums*
 $(1 / (1 - 1 / q) + (- (1 / q) - (1 / q) \wedge p' - (1 / q) \wedge n' - 1))$
by (*intro* *sums-If-finite-set*[*OF* *geometric-sums*])
(use \langle *prime* p' \rangle *q-gt-1* *prime-gt-Suc-0-nat*[*of* p'] \langle $n' > 1$ \rangle **in** \langle *auto* *simp:* \rangle \rangle
also have $\dots = 1 / (q * (q - 1)) - 1 / q \wedge p' - 1 / q \wedge n'$
using *q-gt-1* **by** (*simp* *add: field-simps*)
finally show $(\lambda k. \text{if } k \in \{0, 1, p', n'\} \text{ then } 0 \text{ else } (1 / q) \wedge k)$ *sums* \dots .
next
show $\bigwedge k. (\text{if } k = 0 \vee \text{int } k\ \text{dvd}\ n \text{ then } 0 \text{ else } 1 / q \wedge k)$
 $\leq (\text{if } k \in \{0, 1, p', n'\} \text{ then } 0 \text{ else } (1 / q) \wedge k)$
using *q-gt-1* p **by** (*auto* *simp: p'-def n'-def* *power-divide*)

qed
also have $\dots < 1 / (q * (q - 1)) - 1 / q ^ n'$
using *q-gt-1* **by** *simp*
finally show *?thesis* **by** (*simp add: n'-def N-1*)
qed

This implies that one can use the norm as a primality test:

lemma *prime-iff-N-eq*:
assumes $n \neq 0$
shows $\text{prime-elem } n \longleftrightarrow N\ n = N\ 1 - 1 / q ^ \text{nat } |n|$
proof –
have *: $\text{prime-elem } n \longleftrightarrow N\ n = N\ 1 - 1 / q ^ \text{nat } |n|$ **if** $n > 0$ **for** n
proof –
consider $n = 1 \mid \text{prime } n \mid \neg \text{prime } n\ n > 1$
using $\langle n > 0 \rangle$ **by** *force*
thus *?thesis*
proof *cases*
assume $n = 1$
thus *?thesis* **using** *q-gt-1*
by (*auto simp: N-1*)
next
assume $n: \neg \text{prime } n\ n > 1$
with *nonprime-imp-N-lt*[*of n*] **show** *?thesis* **by** *simp*
qed (*auto simp: N-prime prime-ge-0-int*)
qed

show *?thesis*
proof (*cases n > 0*)
case *True*
with * **show** *?thesis* **by** *blast*
next
case *False*
with *[*of -n*] *assms* **show** *?thesis* **by** *simp*
qed
qed

Factorials, on the other hand, have very small norms:

lemma *N-fact-le*: $N\ (\text{fact } m) \leq 1 / (q - 1) * 1 / q ^ m$
proof (*rule sums-le*[*OF - N-sums*])
have $(\lambda k. 1 / q ^ k / q ^ \text{Suc } m)\ \text{sums } (q / (q - 1) / q ^ \text{Suc } m)$
using *geometric-sums*[*of 1 / q*] *q-gt-1*
by (*intro sums-divide*) (*auto simp: field-simps*)
also have $(q / (q - 1) / q ^ \text{Suc } m) = 1 / (q - 1) * 1 / q ^ m$
using *q-gt-1* **by** (*simp add: field-simps*)
also have $(\lambda k. 1 / q ^ k / q ^ \text{Suc } m) = (\lambda k. 1 / q ^ (k + \text{Suc } m))$
using *q-gt-1* **by** (*simp add: field-simps power-add*)
also have $\dots = (\lambda k. \text{if } k + \text{Suc } m \leq m \text{ then } 0 \text{ else } 1 / q ^ (k + \text{Suc } m))$
by *auto*
finally have $\dots \text{sums } (1 / (q - 1) * 1 / q ^ m)$.

```

also have ?this  $\longleftrightarrow$  ( $\lambda k. \text{if } k \leq m \text{ then } 0 \text{ else } 1 / q^{\wedge} k$ ) sums (1 / (q - 1) *
1 / q^{\wedge} m)
  by (rule sums-zero-iff-shift) auto
finally show ... .
next
fix k :: nat
have int k dvd fact m if k > 0 k ≤ m
proof -
  have int k dvd int (fact m)
    unfolding int-dvd-int-iff using that by (simp add: dvd-fact)
  thus int k dvd fact m
    unfolding of-nat-fact by simp
qed
thus (if k = 0  $\vee$  int k dvd fact m then 0 else 1 / q^{\wedge} k) ≤
  (if k ≤ m then 0 else 1 / q^{\wedge} k) using q-gt-1 by auto
qed

```

```

lemma N-prime-mono:
  assumes prime p prime p' p ≤ p'
  shows N p ≤ N p'
  using assms q-gt-1 by (auto simp add: N-prime field-simps nat-le-iff prime-ge-0-int)

```

```

lemma N-prime-ge:
  assumes prime p
  shows N p ≥ 1 / (q2 * (q - 1))
proof -
  have 1 / (q^{\wedge} 2 * (q - 1)) = N 2
    using q-gt-1 by (auto simp: N-prime N-1 field-simps power2-eq-square)
  also have ... ≤ N p
    using assms by (intro N-prime-mono) (auto simp: prime-ge-2-int)
  finally show ?thesis .
qed

```

```

lemma N-prime-elem-ge:
  assumes prime-elem p
  shows N p ≥ 1 / (q2 * (q - 1))
proof (cases p ≥ 0)
  case True
    with assms N-prime-ge show ?thesis by auto
  next
  case False
    with assms N-prime-ge[of -p] show ?thesis by auto
qed

```

Next, we use this norm to derive a metric:

```

lift-definition dist :: fbint  $\Rightarrow$  fbint  $\Rightarrow$  real is
   $\lambda x y. N (x - y)$  .

```

```

lemma dist-self [simp]: dist x x = 0

```

by transfer simp

lemma *dist-sym* [simp]: $\text{dist } x \ y = \text{dist } y \ x$
by transfer (simp add: N-minus-commute)

lemma *dist-pos*: $x \neq y \implies \text{dist } x \ y > 0$
by transfer (use N-pos in simp)

lemma *dist-eq-0-iff* [simp]: $\text{dist } x \ y = 0 \iff x = y$
using *dist-pos*[of $x \ y$] by (cases $x = y$) auto

lemma *dist-triangle-ineq*: $\text{dist } x \ z \leq \text{dist } x \ y + \text{dist } y \ z$
proof transfer
fix $x \ y \ z :: \text{int}$
show $N \ (x - z) \leq N \ (x - y) + N \ (y - z)$
using *N-triangle-ineq*[of $x - y \ y - z$] by simp
qed

Lastly, we show that the metric we defined indeed induces the Furstenberg topology.

theorem *dist-induces-open*:
 $\text{open } U \iff (\forall x \in U. \exists e > 0. \forall y. \text{dist } x \ y < e \implies y \in U)$

proof (transfer, safe)
fix $U :: \text{int set}$ and $x :: \text{int}$
assume *: $\forall x \in U. \exists b > 0. \text{arith-prog } x \ b \subseteq U$
assume $x \in U$
with * obtain b where $b > 0 \text{ arith-prog } x \ b \subseteq U$ by blast
define e where $e = 1 / q \wedge b$

show $\exists e > 0. \forall y. N \ (x - y) < e \implies y \in U$

proof (rule exI; safe?)

show $e > 0$ using *q-gt-1* by (simp add: *e-def*)

next

fix y assume $N \ (x - y) < e$

also have $\dots = 1 / q \wedge b$ by fact

finally have $b \ \text{dvd} \ (x - y)$

by (rule *N-lt-imp-dvd*) fact

hence $y \in \text{arith-prog } x \ b$

by (auto simp: *arith-prog-def cong-iff-dvd-diff dvd-diff-commute*)

with b show $y \in U$ by blast

qed

next

fix $U :: \text{int set}$ and $x :: \text{int}$

assume *: $\forall x \in U. \exists e > 0. \forall y. N \ (x - y) < e \implies y \in U$

assume $x \in U$

with * obtain e where $e > 0 \ \forall y. N \ (x - y) < e \implies y \in U$ by blast

have eventually $(\lambda N. 1 / (q - 1) * 1 / q \wedge N < e)$ at-top

```

    using q-gt-1 ⟨e > 0⟩ by real-asymp
  then obtain m where m: 1 / (q - 1) * 1 / q ^ m < e
    by (auto simp: eventually-at-top-linorder)
  define b :: nat where b = fact m

  have arith-prog x b ⊆ U
  proof
    fix y assume y ∈ arith-prog x b
    show y ∈ U
    proof (cases y = x)
      case False
      from ⟨y ∈ arith-prog x b⟩ obtain n where y: y = x + int b * n
        by (auto simp: arith-prog-altdef)
      from y and ⟨y ≠ x⟩ have [simp]: n ≠ 0 by auto
      have N (x - y) = N (int b * n) by (simp add: y)
      also have ... ≤ N (int b)
        by (rule N-dvd-mono) auto
      also have ... ≤ 1 / (q - 1) * 1 / q ^ m
        using N-fact-le by (simp add: b-def)
      also have ... < e by fact
      finally show y ∈ U using e by auto
    qed (use ⟨x ∈ U⟩ in auto)
  qed
  moreover have b > 0 by (auto simp: b-def)
  ultimately show ∃ b > 0. arith-prog x b ⊆ U
    by blast
  qed
end

```

We now show that the Furstenberg space is a metric space with this metric (with $q = 2$), which essentially only amounts to plugging together all the results from above.

```

interpretation fb: fbnorm 2
  by standard auto

```

```

instantiation fbint :: dist
begin

```

```

definition dist-fbint where dist-fbint = fb.dist

```

```

instance ..

```

```

end

```

```

instantiation fbint :: uniformity-dist
begin

```

definition *uniformity-fbint* :: (fbint × fbint) filter **where**
uniformity-fbint = (INF e ∈ {0 <..}. principal {(x, y). dist x y < e})

instance by *standard* (simp add: *uniformity-fbint-def*)

end

instance *fbint* :: open-uniformity

proof

fix *U* :: fbint set

show open *U* = (∀ x ∈ *U*. eventually (λ(x',y). x' = x → y ∈ *U*) *uniformity*)

unfolding *eventually-uniformity-metric dist-fbint-def*

using *fb.dist-induces-open* by *simp*

qed

instance *fbint* :: metric-space

by *standard* (use *fb.dist-triangle-ineq* in ⟨*auto simp: dist-fbint-def*⟩)

In particular, we can now show that the sequence $n!$ tends to 0 in the Furstenberg topology:

lemma *tendsto-fbint-fact*: (λn. fbint (fact n)) → fbint 0

proof –

have (λn. dist (fbint (fact n)) (fbint 0)) → 0

proof (rule *tendsto-sandwich*[*OF* *always-eventually* *always-eventually*]; *safe?*)

fix *n* :: nat

show dist (fbint (fact n)) (fbint 0) ≤ 1 / 2 ^ n

unfolding *dist-fbint-def* by (transfer *fixing*: n) (use *fb.N-fact-le*[of n] in *simp*)

show dist (fbint (fact n)) (fbint 0) ≥ 0

by *simp*

show (λn. 1 / 2 ^ n :: real) → 0

by *real-asymp*

qed *simp-all*

thus *?thesis*

using *tendsto-dist-iff* by *metis*

qed

end

References

- [1] A. Dirmeier. On metrics inducing the Furstenberg topology on the integers. <https://arxiv.org/abs/1912.11663>, 2019.
- [2] H. Furstenberg. On the infinitude of primes. *The American Mathematical Monthly*, 62(5):353, May 1955.

- [3] F. Zulfeqarr. Some interesting consequences of Furstenberg topology.
Resonance, 24(7):755–765, July 2019.