DTU Library

Proving in the Isabelle Proof Assistant that the Set of Real Numbers is not Countable

Villadsen, Jørgen

Publication date:
2018

Document Version
Publisher's PDF, also known as Version of record

Proving in the Isabelle Proof Assistant that the Set of Real Numbers is not Countable

Jørgen Villadsen
DTU Compute, AlgoLoG, Technical University of Denmark, 2800 Kongens Lyngby, Denmark
jovi@dtu.dk

Abstract

We present a new succinct proof of the uncountability of the real numbers – optimized for clarity – based on the proof by Benjamin Porter in the Isabelle Analysis theory.

1 Introduction

In 1874 Georg Cantor proved that set of real numbers is not countable – or, no surjective function from the natural numbers to the real numbers exists.

```
theorem ∄f :: nat ⇒ real. surj f
```

We use the Isabelle proof assistant, more precisely Isabelle/HOL, and omit the so-called cartouches ‹...› around formulas as is common in recent papers about formalizations in Isabelle. Since the notion of the real numbers in Isabelle is not grounded in decimal expansions, Cantor's elegant diagonal argument from 1891 is not suitable. With some effort we have ordered by year the immediately known formalizations of the theorem.

ProofPower	Rob Arthan	2003
Metamath	Norman Megill	2004
Mizar	Grzegorz Bancerek	2004
HOL Light	John Harrison	2005
Isabelle	Benjamin Porter	2005
Coq	Nickolay Shmyrev	2006

Freek Wiedijk's comprehensive list "Formalizing 100 Theorems" has been a valuable starting point:

http://www.cs.ru.nl/~freek/100/

We present a new succinct proof – optimized for clarity – based on the proof by Benjamin Porter in the Isabelle Analysis theory and inspired by the traditional proof (Hansen 1999, p. 45). The full proof is available in the appendix and also online here together with other results about countable and uncountable sets:

https://github.com/logic-tools/continuum

We note that the theorem can also be phrased as follows using quantifiers only.

```
proposition ∄f. ∀y :: real. ∃x :: nat. y = f x
```

We have not yet fully investigated if our approach can be generalized to other proofs except that we have recently considered a related proof, namely that the set of rational numbers is in fact countable, based on the rather scattered formalization in the Isabelle Library which incidentally differs in a number of ways from the traditional proof (Hansen 1999).

2 A Possible New Feature in Isabelle

As a possible new feature in Isabelle we use "..." to signify a proof found by Isabelle's Sledgehammer tool (Blanchette 2017), possibly also using some more or less obvious proof methods.

We suggest to implement it like a kind of extended "sorry" proof methods that is a "fake proof pretending to solve the pending claim without further ado" (cf. the Isabelle/Isar Reference Manual in the Isabelle distribution).

But when the Sledgehammer tool finds a proof then the "..." should somehow change color and/or shape to indicate this.

In this way Isabelle proofs can still be replayed.

Perhaps the "..." notation is not ideal since it is used for other things in Isabelle.

3 The Proof Skeleton

We provide a proof skeleton and continue the proof in the following section.

The proof is by contradiction.

```
assume ∃f :: nat ⇒ real. surj f
show False
```

We first obtain a name for the surjective function.

```
from ∃f. surj f obtain f :: nat ⇒ real where surj f ..
then have assumption: ∃n. f n = z for z ...
```

Here ".." is a standard proof; it abbreviates "by standard" and performs elementary proof steps depending on the application environment. And the "..." proof is a resolution proof "by (metis surj_def)" which we for further transparency separate into two proof steps "unfolding surj_def by metis" as shown in the appendix.

In our proof we now obtain a certain natural-numbers-indexed set D of real numbers with a kind of diagonalization property.

```
obtain D :: nat ⇒ real set
  where
    (⋂n. D n) ≠ {}
    f n ∉ D n
  for n
```

We defer the proof of the existence of the indexed set D to the next section. From the indexed set D we easily obtain the contradiction.

```
then obtain e where ∄n. f n = e ...
moreover from assumption have ∃n. f n = e .
ultimately show ?thesis ..
```

Here "…" is the resolution proof "by (metis INT_E UNIV_I ex_in_conv)" as shown in the appendix.

4 The Indexed Set D

We need to fill the gap in the proof skeleton regarding the indexed set D.

We start by defining two functions of three arguments.

```
obtain L R :: real ⇒ real ⇒ real ⇒ real
  where *:
    L a b c < R a b c
    {L a b c .. R a b c} ⊆ {a .. b}
    c ∉ {L a b c .. R a b c}
  if a < b for a b c
```

We here include the complete proof of the existence of the two functions, except for the "..." proofs shown in the appendix.

```
proof -
  have ∃x y. a ≤ x ∧ x < y ∧ y ≤ b ∧ ¬ (x ≤ c ∧ c ≤ y)
    if a < b for a b c :: real ...
  then have ∃x y. x < y ∧ {x .. y} ⊆ {a .. b} ∧ c ∉ {x .. y}
    if a < b for a b c :: real ...
  then show ?thesis ...
qed
```

We recursively define an indexed set of intervals given by pairs – the endpoints of the intervals.

```
define P :: nat ⇒ real × real
  where
    P ≡ rec_nat
        (L 0 1 (f 0),
         R 0 1 (f 0))
        (λn (x, y). (L x y (f (Suc n)),
                     R x y (f (Suc n))))
```

We prove that the endpoints are ordered as expected; again the "..." proofs are shown in the appendix.

```
with *(1) have 0: fst (P n) < snd (P n) for n ...
```

Finally we define the indexed set of intervals and prove the required properties.

```
define I :: nat ⇒ real set
  where
    I ≡ λn. {fst (P n) .. snd (P n)}
with 0 have I n ≠ {} for n ...
moreover from 0 *(2) have decseq I ...
ultimately have finite S ⟶ (⋂n∈S. I n) ≠ {} for S ...
moreover have closed (I n) for n ...
moreover have compact (I n) for n ...
ultimately have (⋂n. I n) ≠ {} ...
moreover from 0 *(3) have f n ∉ I n for n ...
ultimately show ?thesis ..
```

5 Conclusion

We have with good results explained the proof to a group of mathematicians with little or no knowledge of formal methods. In particular the "…" notation is useful and might be relevant to implement, perhaps with the Proof Strategy Language available in the Isabelle Archive of Formal Proofs.

References

Vagn Lundsgaard Hansen (1999): *Fundamental Concepts in Modern Analysis*. World Scientific.
Jasmin Christian Blanchette (2017): *User's Guide to Sledgehammer*. Isabelle Distribution.

Appendix: Formalization in Isabelle

```
theory Scratch imports Complex_Main
begin

theorem ‹∄f :: nat ⇒ real. surj f›
proof
  assume ‹∃f :: nat ⇒ real. surj f›
  show False
  proof -
    from ‹∃f. surj f› obtain f :: ‹nat ⇒ real› where ‹surj f› ..

    then have assumption: ‹∃n. f n = z› for z
      unfolding surj_def by metis
```

```
    obtain D :: ‹nat ⇒ real set› where ‹(⋂n. D n) ≠ {}› ‹f n ∉ D n› for n
    proof -
      obtain L R :: ‹real ⇒ real ⇒ real ⇒ real›
        where
          *: ‹L a b c < R a b c› ‹{L a b c .. R a b c} ⊆ {a .. b}› ‹c ∉ {L a b c .. R a b c}›
        if ‹a < b› for a b c
      proof -
        have ‹∃x y. a ≤ x ∧ x < y ∧ y ≤ b ∧ ¬ (x ≤ c ∧ c ≤ y)› if ‹a < b› for a b c :: real
          using that dense less_le_trans not_le not_less_iff_gr_or_eq by (metis (full_types))

        then have ‹∃x y. x < y ∧ {x .. y} ⊆ {a .. b} ∧ c ∉ {x .. y}› if ‹a < b› for a b c :: real
          using that by fastforce

        then show ?thesis
          using that by metis
      qed

      define P :: ‹nat ⇒ real × real›
        where
          ‹P ≡ rec_nat
               (L 0 1 (f 0),
                R 0 1 (f 0))
               (λn (x, y). (L x y (f (Suc n)),
                            R x y (f (Suc n))))›

      with *(1) have 0: ‹fst (P n) < snd (P n)› for n
        unfolding split_def by (induct n) simp_all

      define I :: ‹nat ⇒ real set›
        where
          ‹I ≡ λn. {fst (P n) .. snd (P n)}›

      with 0 have ‹I n ≠ {}› for n
        using less_imp_le by fastforce

      moreover from 0 *(2) have ‹decseq I›
        unfolding I_def P_def split_def decseq_Suc_iff by simp

      ultimately have ‹finite S ⟶ (⋂n∈S. I n) ≠ {}› for S
        using decseqD subset_empty INF_greatest Max_ge by metis

      moreover have ‹closed (I n)› for n
        unfolding I_def by simp

      moreover have ‹compact (I n)› for n
        unfolding I_def using compact_Icc compact_Int_closed decseqD inf.absorb_iff2 le0 by simp

      ultimately have ‹(⋂n. I n) ≠ {}›
        using INT_insert compact_imp_fip_image empty_subsetI finite_insert inf.absorb_iff2 by metis

      moreover from 0 *(3) have ‹f n ∉ I n› for n
        unfolding I_def P_def split_def by (induct n) simp_all

      ultimately show ?thesis ..
    qed

    then obtain e where ‹∄n. f n = e›
      using INT_E UNIV_I ex_in_conv by metis

    moreover from assumption have ‹∃n. f n = e› .

    ultimately show ?thesis ..
  qed
qed

end — ‹Jørgen Villadsen, DTU Denmark - Based on work by Benjamin Porter, NICTA Australia›
```