

Formalizing a Discrete Model of the Continuum in Coq from a Discrete Geometry Perspective

Nicolas Magaud, Agathe Chollet, Laurent Fuchs

▶ **To cite this version:**

Nicolas Magaud, Agathe Chollet, Laurent Fuchs. Formalizing a Discrete Model of the Continuum in Coq from a Discrete Geometry Perspective. Annals of Mathematics and Artificial Intelligence, 2015, 10.1007/s10472-014-9434-6 . hal-01066671

HAL Id: hal-01066671

https://inria.hal.science/hal-01066671

Submitted on 22 Sep 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Formalizing a Discrete Model of the Continuum in Coq from a Discrete Geometry Perspective

Nicolas Magaud[1], Agathe Chollet[2], Laurent Fuchs[3]

[1] Icube UMR 7357 CNRS - Université de Strasbourg
300 Bld Sébastien Brant BP 10413 67412 Illkirch Cedex, France
magaud@unistra.fr,
[2] Laboratoire MIA - Université de La Rochelle,
Avenue Michel Crépeau 17042 La Rochelle Cedex, France
achollet01@univ-lr.fr,
[3] Laboratoire XLIM-SIC UMR 6172 CNRS - Université de Poitiers
Bat. SP2MI Bld Marie et Pierre Curie
BP 30179 86962 Futuroscope Chasseneuil Cedex, France
Laurent.Fuchs@sic.univ-poitiers.fr.

Abstract. This work presents a formalization of the discrete model of the continuum introduced by Harthong and Reeb [16], the Harthong-Reeb line. This model was at the origin of important developments in the Discrete Geometry field [31]. The formalization is based on the work presented in [8,7] where it was shown that the Harthong-Reeb line satisfies the axioms for constructive real numbers introduced by Bridges [4]. Laugwitz-Schmieden numbers [20] are then introduced and their limitations with respect to being a model of the Harthong-Reeb line is investigated [7]. In this paper, we transpose all these definitions and properties into a formal description using the Coq proof assistant. We also show that Laugwitz-Schmieden numbers can be used to actually compute continuous functions. We hope that this work could improve techniques for both implementing numeric computations and reasoning about them in geometric systems.

1 Introduction

Dealing with geometric problems (geometric constraints solving, geometric modeling) people are, finally, faced with computations that involve computer representation of real numbers. Due to their important impact, the studies about real numbers in computer science are numerous and our purpose is not to surpass them but to reactivate an effective point of view that has been forgotten for a while [31].

This point of view was built in the eighties by J. Harthong and G. Reeb [16] and consists in a model of the continuum based over the integers that is the Harthong-Reeb line. This model was at the origin of important developments in the Discrete Geometry field [31]. And, at that time, the constructive content of this model was neglected even though it was explicitly noted in Diener and

Reeb's book [11]. As the Harthong-Reeb line is at the origin of the development of numerous and significant results on discrete straight lines and discrete circles, we hope that the present work will improve the confidence we can have in theoretical results and algorithms in the discrete analytical geometry field. Moreover, the Harthong-Reeb line based on Ω-integers opens the possibility to study the properties of multiscale objects in discrete analytical geometry. So our formalization can also be a support for those studies.

In previous works [8] it was shown that the Harthong-Reeb line satisfies the axioms for constructive real numbers introduced by Bridges [4]. However, the Harthong-Reeb line construction is based on a nonstandard arithmetic of the integers that was not explicitly built. To be short, starting with the naive integer sequence (the one that you can enumerate: 1, 2, ...), G. Reeb argues that there must exist an integer ω that is greater than all naive integers. Using the compactness theorem[4] from model theory [17], the existence of this nonstandard integer ω is sufficient to deduce that there exists a nonstandard model of integer arithmetic with such nonstandard integer ω and then this model can be used to build the Harthong-Reeb line.

Nevertheless, this nonstandard model of integer arithmetic is not built and, in order to be put on computers, the Harthong-Reeb line needs a constructive nonstandard model of integer arithmetic. Such a construction, based on the Ω-numbers of Laugwitz and Schmieden [21], was made by some of the authors with others in [9,7].

This work presents a formalization of the Harthong-Reeb line using the Coq proof assistant. It can be seen as a light counterpart of the seminal works about the formalization of exact arithmetic [15,30,19]. Our motivations to do this work come from the difficulties that we faced when showing that the Harthong-Reeb line satisfies the axioms proposed by Bridges [4]. Unless proofs have been read carefully we have no way to be sure that they were entirely correct. This confidence problem of proofs is mainly due to the unusual mathematics that we deal with. The arithmetic dealt with is in a nonstandard framework and the axioms are in a constructive framework. So, it was not clear that handwritten proofs did not contain subtle mistakes or imprecisions. Moreover, the formalization has entailed a better understanding of how concepts and proofs are related to one another.

From a more practical point of view, the Harthong-Reeb line provides a rich theoretical framework that allows us to analyze a wide range of geometrical objects. So, our formalization can also be thought as a computable model for geometric computations. One main advantage of such model is that computing with these algorithms and reasoning about them (e.g. to prove that they are correct) can be done in the same framework. And we hope that this will help developing geometric systems where computations are made using the Harthong-Reeb line. This is possible, as shown in this paper, because the Coq proof assistant [10,1] implements a higher order constructive logic and is also a programming language

[4] Roughly speaking it says that if for a theory with infinitely many axioms, each finite subset of axioms has a model then the theory has a model.

equipped with inductive definitions and recursive functions. Therefore, it is the perfect tool to carry out a constructive formalization.

Let us summarize our overall strategy and expectations: we aim at describing in Coq the Harthong-Reeb line, in a way which allows us to compute continuous functions as well as to prove its properties formally. We adopt the point of view of Bridges in [4,2] where constructive mathematics are those which use intuitionistic logic only. To build the Harthong-Reeb line, we first consider an abstract (axiomatic) representation of nonstandard integers. We prove that the Harthong-Reeb line built on top of it actually verifies the axioms of Bridges for being a constructive real line. Then we try and find an actual model of these nonstandard integers, and focus on Laugwitz-Schmieden integers. Although this implementation does not allow us to prove that all axioms of Bridges hold (we can only build an alternative version of the continuum and some of Bridges' axioms need to be slightly modified), it provides us with a framework in which one can compute continuous functions and it opens some interesting perspectives for multiscale representation of more complex geometric objects (circles, etc.).

This paper is organized as follows. In section 2, we discuss related work. In section 3, we formally describe in Coq[5] what a nonstandard model of arithmetic should be. In Section 4, we build the Harthong-Reeb line $\mathcal{HR}_\omega$ on top of it and prove that $\mathcal{HR}_\omega$ verifies Bridges' Axioms which capture what a constructive real line is. In Section 5, we investigate how the Ω-numbers of Laugwitz and Schmieden can be an adequate model of the nonstandard arithmetic we consider and show that we can compute continuous fonctions using Euler's arithmetization scheme. Finally, in Section 6, we discuss our results as well as future work.

2 Related Work

We first reported on our experiments with formalizing the Harthong-Reeb line in Coq at the ADG'2010 workshop [24]. It led to an alternative formalization in Isabelle/HOL [13]. The two formalizations are independent and do not rely on the same foundations. Fleuriot uses the existing mechanization of nonstandard analysis available in Isabelle/HOL. His formalization is based on hyperreals, which are an extension of the real numbers $\mathbb{R}$ obtained using the so-called ultrapower construction [14]. This construction happens to be not constructive.

In our formal developement in Coq, we follow the layout of previous papers on the Harthong-Reeb line [8,7] and we consider a minimal axiomatic form of nonstandard analysis, related to Nelson' s Internal Set Theory (IST) [28]. We also intend to capture the inheritely constructive nature of the model of the Harthong-Reeb line in our formalization, using Ω-numbers, introduced by Laugwitz and Schmieden in [22]. Indeed, the aim, in addition to proving formally some properties of the Harthong-Reeb line is to be able to derive actual (Ocaml) programs from the formalization of operations such as Euler's arithmetization scheme on Ω-numbers. Such programs can be directly written in Coq, and thus

[5] Most statements in Coq are self-explanatory, when it becomes more technical, detailed explanations will be given.

one can both compute using these programs and formally prove some properties about their behavior.

There exists alternative models of nonstandard analysis, e.g. [18,27]. However, we are interested in building a model which is easy to implement and practically usable to perform computations in the context of discrete analytical geometry. That is why we focus on Laugwitz-Schmieden Ω-numbers for our concrete implementation of nonstandard numbers. Indeed, such a model makes it possible to have a multi-scale representation of geometric objects.

Our research is application-driven and we expect these new results to help improve the quality of sofware used in the field of discrete analytical geometry. We did not investigate some more theoretical issues such as links with Tennenbaum's theorem; we only aim at providing a better representation of numbers (w.r.t floating-point numbers) for applications such as computations of continuous functions.

3 A minimal axiom system for nonstandard arithmetic

The ground idea of the Harthong-Reeb line is to introduce a non trivial rescaling on the set of integers in order to get a discrete form of the continuum. To do so a nonstandard arithmetic is used. In this section, we present a formalization of such a nonstandard arithmetic on top of which we shall build the Harthong-Reeb line.

3.1 Nonstandard Model of Arithmetic

Axiomatizing Numbers We first have to specify the axiomatic numbers we shall use in this work as well as their operations and their properties. We do that using a module type in Coq. From a programmming perspective, this can be viewed as an interface which, on the one hand, gathers all the properties of nonstandard numbers required to build the Harthong-Reeb line and on the other hand, can be implemented by a concrete datatype, operations and proofs of the axioms (as we do in section 5). This module type contains the declaration of the basic objects of the theory (A is the abstract data type denoting non standard integers):

```
Parameter A:Type.

Parameter a0 a1 : A.
Parameter plusA multA divA modA : A -> A -> A.
Parameter oppA absA : A -> A.

Parameter leA ltA : A -> A -> Prop.
```

We choose to let A live in `Type`, which is the topmost Sort in Coq. The Coq proof assistant features three sorts `Prop`, `Set` and `Type`. Usually, `Prop` is the type of propositions, whereas `Set` is the type of computations. Moreover, `Type`

is the type of `Prop` and `Set`. For convenience, we choose to have A : `Type`, as the axiomatic real numbers of Coq also live in `Type`.

Notations can be introduced to ease reading and writing of specifications. This also allows us to stay close to the way mathematicians would write.

```
Notation "x + y " := (plusA  x y).
Notation "x * y " := (multA  x y).
Notation "x / y " := (divA  x y).

Notation "0" := (a0).
Notation "1" := (a1).
Notation "- x" := (oppA x).
Notation "| x |" := (absA x) (at level 60).
Notation "x <= y" := (leA x y) (at level 50).
Notation "x < y" := (ltA x y) (at level 50).
```

Then all the basic properties of A are expressed as axioms.

```
Parameter plus_neutral : forall x,0 + x = x.
Parameter plus_comm : forall x y,  x + y =  y + x.
Parameter plus_assoc : forall x y z,  x + (y + z) = (x + y) + z.
Parameter plus_opp : forall x, x + (- x) = 0.

Parameter abs_pos : forall x, 0 <=|x|.
Parameter abs_pos_val : forall x, 0 <=x -> |x|=x.
Parameter abs_neg_val : forall x, x <=0 -> |x|=-x.
[...]
```

Overall, we assume that A with the operations $+, \times, \ldots$ is equipped with a ring structure. This will allow to prove basic algebraic equations automatically and also to perform some otherwise tedious simplifications of expressions. We also assume that it is equipped with an euclidian division operator and the absolute value ($x \mapsto |x|$).

In addition, we assume that the order relations $\leq$ and $<$ enjoy their usual properties such as transitivity, regularity w.r.t operations such as addition, etc. We also assume these relations are decidable by adding the following axiom which states that forall $x : A$, either $x < 0$ or $x = 0$ or $0 < x$.

```
Axiom A0_dec : forall x, {x < 0}+{x = 0}+{0 < x}.
```

Nonstandard aspects Even if axiomatizations of nonstandard analysis, such as IST [28], are available, we present here, in the spirit of some works of Nelson or Lutz [29,23], a weaker axiom system which is well suited for our purpose. First we introduce a new predicate `lim` over integer numbers: `lim`(x) "means" that the integer x is limited. This intends to capture the idea that x is not infinitely large. We also introduce the number ω denoted by `w`. This number is larger than all naive integers as argued by Reeb in [11].

```
Parameter lim : A -> Prop.
Parameter w : A.
```

This predicate is external to the classical integer theory and its meaning directly derives from the following axioms ANS1, ANS2, ANS3, ANS4 (and ANS5 which will be introduced later):

ANS1. *The number* 1 *is limited.*

```
Parameter ANS1 : lim 1.
```

ANS2. *The sum and the product of two limited numbers are limited.*

```
Parameter ANS2a : forall x y, lim x -> lim y -> lim (x + y).
Parameter ANS2b : forall x y, lim x -> lim y -> lim (x * y).
```

ANS3. *Non-limited integer numbers exist.*

```
Parameter ANS3 : ~ lim w.
```

We simply assert that w is not limited (in Coq, `~` stands for logic negation).

ANS4. *For all* $(x, y) \in A^2$ *such that* x *is limited and* $|y| \leqslant |x|$*, the number* y *is limited.*

```
Parameter ANS4 :
  forall x, (exists y, lim y /\ | x | <= | y |)-> lim x.
```

For reading conveniences, we introduce the following notations [8]:

- $\forall^{lim} x\ F(x)$ is an abbreviation for $\forall x\ (lim(x) \Rightarrow F(x))$ and can be read as "for all limited x, $F(x)$ stands".
- $\exists^{lim} x\ F(x)$ is an abbreviation for $\exists x\ (lim(x) \wedge F(x))$ and can be read as "there exists a limited x such that $F(x)$".

Following the IST flavour [28], we say that a formula or a proposition P is *external* when the predicate `lim` occurs in P and *internal* otherwise. This distinction is necessary to determine when properties known for standard properties can be extended to the nonstandard ones. In fact, when a property P is internal, i.e. when it does not use the predicate `lim`, the extension of P to infinitely large numbers is immediate. This is given by the following *Overspill principle*. But for external properties, we cannot proceed in the same way. We need to introduce a new extension principle as an axiom (called ANS5 in this paper). This principle states that the formula which contains external properties can be extended to infinitely large numbers, but that we do not know whether these infinitely large numbers verify these properties.

Proposition 1. ***(Overspill principle)*** *Let* $\mathcal{P}(x)$ *be an internal formula such that* $\mathcal{P}(n)$ *is true for all* $n \in A$, $\texttt{lim}n \wedge n \geqslant 0$*. Then, there exists an infinitely large*[6] $\nu \in A, \nu \geqslant 0$ *such that* $\mathcal{P}(m)$ *is true for all integers* m *such that* $0 \leqslant m \leq \nu$*.*

[6] An *infinitely large* number is the same as a *non-limited* number.

```
Parameter overspill_principle : forall P:A -> Prop,
 (forall n:A, lim n /\ 0<=n -> P n) ->
  (exists v:A, ~lim v /\ 0<=v /\ (forall m:A, 0<=m /\ m <=v -> P m)).
```

The following principle, which is our last axiom, can deal with both an *internal* and an *external* formula P.

ANS5. ***(External inductive defining principle):*** *We suppose that*

1. *there is $x_0 \in A^p$ such that $\mathcal{P}((x_0))$;*
2. *for all $n \in A, n \geqslant 0 \wedge lim(n)$ and all sequence $(x_k)_{0\leqslant k\leqslant n}$ in A^p such that $\mathcal{P}((x_k)_{0\leqslant k\leqslant n})$ there is $x_{n+1} \in A^p$ such that $\mathcal{P}((x_k)_{0\leqslant k\leqslant n+1})$.*

Then, there exists an internal sequence $(x_k)_{k\in A,k\geqslant 0}$ in A^p such that, for all n which verifies $limn \wedge n \geqslant 0$, we have $\mathcal{P}((x_k)_{0\leqslant k\leqslant n})$.

This principle means that the sequence of values x_k for k limited can be prolonged in an infinite sequence $(x_k)_{k\in A,k\geqslant 0}$ defined for all integers. Saying that this sequence is internal means that it has all the properties of the classical sequences in usual number theory. In particular, if $\mathcal{Q}(x)$ is an internal formula, then the set $\{k \in A, k \geqslant 0\ ;\ \mathcal{Q}(x_k)\}$ is an internal part of $\{k \in A, k \geqslant 0\}$.

In Coq, we choose a slighly different and more convenient definition of ANS5. First we only consider the case $p = 1$. In addition, we choose to have a predicate P whose arity is fixed. In the definition of `ANS5` in Coq, the predicate P only applies to a single element of the sequence rather than to the whole sequence. It can be viewed as a projection on the original $\mathcal{P}$ on the k_{th} element of the sequence. This means a statement such as $\mathcal{P}((x_k)_{0\leqslant k\leqslant n})$ is translated into $\forall k, 0 \leqslant k \leqslant n \rightarrow P(x_k)$. Finally, we have to make explicit that the sequence whose existence is shown coincides with the initial one for all k such that $0 \leqslant k \leqslant n$. Overall, this leads to a weaker principle which, at the same time, is sufficient for our purposes and more convenient to use in Coq. Its statement in Coq is the following one:

```
Parameter ANS5 :
forall P :A -> Prop,
(forall u : forall n:A, lim n /\ 0 <= n -> A,
  P (u 0 H0) ->
  (forall n:A, forall Hn : lim n /\ 0 <= n,
     (forall k:A, forall Hk:0 <= k /\ k <= n,
      forall Hn1 : lim (n+1)/\0 <=(n+1),
        P (u k (ANS4_special n k Hn Hk)) -> P(u (plusA n 1) Hn1))) ->
          {v:A->A | forall n:A, forall Hn:lim n /\ 0 <= n,
                    forall k:A,  forall Hk:0 <= k /\ k <= n,
                    P (v k) /\ v k = u k (ANS4_special n k Hn Hk)}).
```

Here, the sequence u expects two arguments: an element n of A and a proof `H` that `lim` $n \wedge n \geqslant 0$. In this context, `H0` is a proof of `lim` $0 \wedge 0 \geqslant 0$ and `(ANS4_special n k Hn Hk)` is a proof of `lim` $k \wedge k \geqslant 0$.

Note that `{x:A|P x}`, which is a convenient notation for existential Σ-types (`sig` P), allows us to describe sets comprehensively. Here it corresponds to the set of elements of A which verify P. This corresponds to an inductive definition in Coq. It comes together with two projections: `proj1_sig`, which returns `a` and `proj2_sig` which returns a proof `H` of `P a`. In addition, `ANS4_special` is a theorem which is derived from `ANS4` and states the following property:

```
Lemma ANS4_special :
 forall n k:A, (lim n /\ 0<=n) -> (0<=k /\ k<=n) -> lim k /\ 0<=k.
```

4 Formalizing the Harthong-Reeb line: A modular approach

4.1 The system $\mathcal{HR}_\omega$.

Let us now give the definition of the system $\mathcal{HR}_\omega$. Introduced by M. Diener [12], this system is the formal version of the so-called Harthong-Reeb line. In this section we prove that this system can be viewed as a model of the real line which is partly constructive. Indeed, we do not have means to construct an infinitely large number ω yet. In some sense, $\mathcal{HR}_\omega$ is equivalent to $\mathbb{R}$ (see [8] for details).

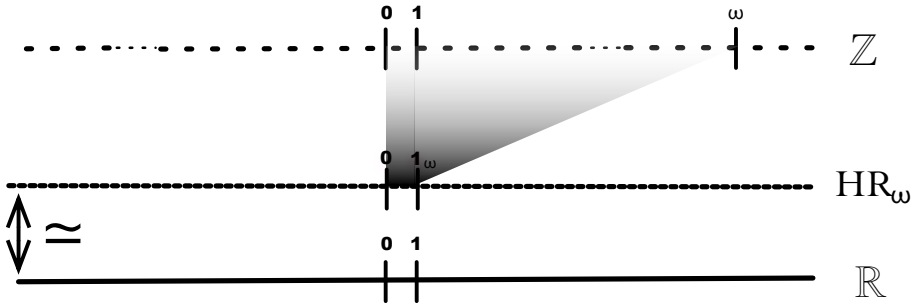


Fig. 1. An intuitive representation of $\mathcal{HR}_\omega$.

Accordingly to axiom ANS3, the construction starts by considering an infinitely large (non-limited) positive integer $\omega \in A$. Our purpose is to define a new numerical system such that all the elements are integers and, in which ω is the new unit. Let us introduce the underlying set of this system.

Definition 1. *The set $\mathcal{HR}_\omega$ of the admissible integers considering the scale ω is defined by:* $\mathcal{HR}_\omega = \{x \in A \ ; \ \exists^{lim} n \in A, n \geqslant 0 \ \text{ such that } |x| < n\omega\}$.

This definition can be easily translated in Coq:

```
Definition P :=
  fun (x:A)=> exists n:A, (lim n /\ 0 < n /\  (|x| <= n*w)).
Definition HRw := {x:A | P x}.
```

The set $\mathcal{HR}_\omega$ is an external set. Moreover, it is an additive sub-group of A. We provide $\mathcal{HR}_\omega$ with the operations $+_\omega$ and $*_\omega$, the ω-scale equality, the ω-scale inequality relations (noted $=_\omega$ and $\neq_\omega$) and the order relation $>_\omega$:

Definition 2. *Let X and Y be any elements of $\mathcal{HR}_\omega$.*

- *X and Y are* equal *at the scale ω and we write $X =_\omega Y$ when* $\forall^{lim} n \in A \quad n > 0 \rightarrow n|X - Y| \leqslant \omega$.
- *Y is* strictly greater *than X at the scale ω and we write $Y >_\omega X$ when* $\exists^{lim} n \in A \quad n > 0 \wedge n\,(Y - X) \geqslant \omega$.
- *X is* different *from Y at the scale ω and we write $X \neq_\omega Y$ when* $(X >_\omega Y$ *or* $Y >_\omega X)$
- *The* sum *of X and Y at the scale ω is $X +_\omega Y := X + Y$ (like the usual sum). For this operation, the* neutral element *is $0_\omega = 0$ and the* opposite *of each element $Z \in \mathcal{HR}_\omega$ is $-_\omega Z := -Z$.*
- *The* product *of X and Y at the scale ω is $X \times_\omega Y := \lfloor \frac{X.Y}{\omega} \rfloor$ (different from the usual one). The* neutral element *is $1_\omega := \omega$, and the* inverse *of each element $Z \in \mathcal{HR}_\omega$ such that $Z \neq_\omega 0_\omega$ is $Z^{(-1)_\omega} := \left\lfloor \frac{\omega^2}{Z} \right\rfloor$.*

Algebraic operations are defined on the integers of the set A onto which $\mathcal{HR}_\omega$ is built. Therefore we must ensure the result still belongs to $\mathcal{HR}_\omega$. For the sum, it consists in proving the following lemma:

```
Lemma Padd: forall x y:A, P x -> P y -> P ( x + y).
```

Then, the addition in $\mathcal{HR}_\omega$ can be defined as follows:

```
Definition HRwplus (x y: HRw) : HRw :=
match x with exist xx Hxx =>
match y with exist yy Hyy =>
exist P (xx + yy) (Padd xx yy Hxx Hyy)
end end.
```

All lemmas and formal definitions of the objects of Definition 2 are summarized in Fig. 2.

4.2 Proving Bridges' Axioms

In the 90's, Brigdes proposed in [4] an axiomatic definition of what a constructive real line is. It is decomposed into three groups about algebraic structure (R1), ordered set (R2) and the last group (R3) deals with special properties (see the details below). A field which satisfies these axioms is called a Bridges-Heyting ordered field. In [8], the proof of that *the Harthong-Reeb line with associated*

```
Lemma Padd : forall x y, P x -> P y -> P (x + y).

Definition HRwplus (x y: HRw) : HRw :=
 match x with exist xx Hxx => match y with exist yy Hyy =>
  exist P (xx + yy) (Padd xx yy Hxx Hyy)
end end.

Lemma Popp : forall x, P x -> P (- x).

Definition HRwopp (x: HRw) : HRw :=
 match x with exist xx Hxx => exist P (- xx) (Popp xx Hxx) end.

Definition HRwminus (x y : HRw) : HRw := HRwplus x (HRwopp y).

Lemma Pprod : forall x y, P x -> P y -> P (( x * y) / w).

Definition HRwmult (x y: HRw) : HRw :=
 match x with exist xx Hxx => match y with exist yy Hyy =>
  exist P ((xx * yy) / w) (Pprod xx yy Hxx Hyy)
end end.

Definition HRwequal (x y : HRw) : Prop :=
 match x with exist xx Hxx => match y with exist yy Hyy =>
  (forall n, lim n ->0 < n -> ( (n*|xx + (- yy)|) <= w))
end end.

Definition HRwgt (y x : HRw) : Prop :=
 match y with exist yy Hyy => match x with exist xx Hxx =>
  (exists n, lim n /\ 0 < n /\ (w <= (n*(yy+ (-xx)))))
end end.

Definition HRwge (y x:HRw) : Prop :=
 match y with exist yy Hyy => match x with exist xx Hxx =>
  forall n, lim n /\ 0 < n -> n * (xx + - yy) <= w
end end.

Definition HRwdiff (x y : HRw) : Prop := HRwgt x y \/ HRwgt y x.

Lemma Pdiv : forall x ,  HRwdiff x HRw0 -> P ((w * w ) /(proj1_sig x)).

Definition HRwinv (x : HRw) (H: HRwdiff x HRw0) : HRw :=
 exist P ((w * w ) / (proj1_sig x)) (Pdiv x H).
```

Fig. 2. Definitions of $\mathcal{HR}_\omega$ operations in Coq

operations and relations is a Bridges-Heyting ordered field is given and it only uses intuitionistic logic.

In this section, we show how the required properties can be expressed and proved correct in Coq.

R1. Algebraic structure $\forall x, y, z \in \mathcal{HR}_\omega$,

1. $x +_\omega y =_\omega y +_\omega x$
2. $(x +_\omega y) +_\omega z =_\omega x +_\omega (y +_\omega z)$
3. $0_\omega +_\omega x =_\omega x$
4. $x +_\omega (-_\omega x) =_\omega 0_\omega$
5. $x \times_\omega y =_\omega y \times_\omega x$
6. $(x \times_\omega y) \times_\omega z =_\omega x \times_\omega (y \times_\omega z)$
7. $1_\omega \times_\omega x =_\omega x$
8. $x \times_\omega x^{(-1)_\omega} =_\omega 1_\omega$ if $x \neq_\omega 0_\omega$
9. $x \times_\omega (y +_\omega z) =_\omega x \times_\omega y +_\omega x \times_\omega z$

This first group presents the expected properties about the two operations $+_\omega$ and $\times_\omega$. There is not any major difficulties to prove that $\mathcal{HR}_\omega$ verifies these axioms.

All the axioms of this group can be formally proved using the definitions of the operations involved. Most of these properties are expressed using Leibniz equality of Coq. They proceed by case analysis on the elements of $\mathcal{HR}_\omega$, destructuring them into an element x of A and a proof H that $P(x)$ holds. We present the proof of the first one (commutativity of addition). Proving the terms `HRwplus` x y and `HRwplus` y x are equal in $\mathcal{HR}_\omega$ consists in not only proving the witnesses (in A) are equal but also proving the proofs of the properties $P(x+y)$ and $P(y+x)$ are equal. As what matters is only that P holds for the considered element, we use the principle of *proof irrelevance* to show all proofs of the same property (e.g. $P(x)$) are equal[7]. This principle is expressed with the following axiom in Coq:

```
Axiom proof_irrelevance : forall (P:Prop) (p1 p2:P), p1 = p2.
```

This well-known principle is consistent with Coq's logic (provided P is in `Prop`) and therefore we can safely add it to our formal description (see [5, Chap. 12] for details).

Some properties such as (6), (8) and (9) do require using the equality $=_\omega$ defined above. Indeed when computations at the level of A involve products and quotients, terms can no longer be shown equal w.r.t Leibniz equality. If necessary, all properties can be established for the $\mathcal{HR}_\omega$ equality $=_\omega$. Indeed, this defined equality can be easily derived from Leibniz equality when needed.

Once all the properties of the first group have been proved in Coq, we can declare $\mathcal{HR}_\omega$ equipped with its operations and the equality relation $=_\omega$ as a ring structure. It works in the same way we did it for A and also allows us to carry out simplifications of algebraic expressions of type $\mathcal{HR}_\omega$. The only restriction is that morphisms such as

$$\forall x, y, z, t \in \mathcal{HR}_\omega \quad x =_\omega y \Rightarrow z =_\omega t \Rightarrow x \times_\omega z =_\omega y \times_\omega t$$

[7] To do that, we use the definitions and axioms from the library `Coq.Logic.ProofIrrelevance`

needs to be proved by hand to be able to do some rewriting.

R2. Basic properties of $>_\omega$ $\forall x, y, z \in \mathcal{HR}_\omega$,

1. $\neg\,(x >_\omega y \wedge y >_\omega x)$
2. $(x >_\omega y) \Rightarrow \forall z\ (x >_\omega z\ or\ z >_\omega y)$
3. $\neg(x \neq_\omega y) \Rightarrow x =_\omega y$
4. $(x >_\omega y) \Rightarrow \forall z\ (x +_\omega z >_\omega y +_\omega z)$
5. $(x >_\omega 0_\omega \wedge y >_\omega 0_\omega) \Rightarrow x \times_\omega y >_\omega 0_\omega$

All these properties can be proved in a very straightforward manner in Coq, following the informal proofs of [8]. Note that this definition of inequality is quite more complex than the usual one. This comes from the fact that the decidability of $>_\omega$ is not necessarily required. Thanks to this definition of inequality on the Harthong-Reeb line, these above-mentioned axioms are easily provable. We just need to assume that the basic inequality on A is decidable. This hypothesis is not a problem for the axiomatic definition of nonstandard arithmetic we consider so far but, in practice there are some problems. We shall work on these issues in Section 5 for the Laugwitz-Schmieden model where this inequality is not decidable. An alternative approach, being currently investigated by G. Wallet, would be to obtain this property using another model based on a non standard variant of Martin-Löf's Type Theory[25,26].

Links between orders in $\mathcal{HR}_\omega$ and orders in A. We recall that, in $\mathcal{HR}_\omega$, the strictly greater relation $>_\omega$ and the greater or equal relation $\geqslant_\omega$ are defined from the less or equal relation on A ($y >_\omega x \equiv \exists^{\lim} n \in A \quad n\,(y - x) \geqslant \omega$ and $y \geqslant_\omega x \equiv \forall^{\lim} n \in A \quad n\,(y - x) \leqslant \omega$). However, we can characterize the $\geqslant_\omega$ relation using the $>_\omega$ relation as follows:

$$\forall a, b \in \mathcal{HR}_\omega \quad b \geqslant_\omega a \equiv \forall c \in \mathcal{HR}_\omega \quad a >_\omega c \Rightarrow b >_\omega c.$$

This makes proofs of properties about $\geqslant_\omega$ much easier to write. Finally, we also have the two following correspondences for all $a, b \in \mathcal{HR}_\omega$:

$$a \geqslant b \text{ implies } a \geqslant_\omega b \quad \text{and} \quad a >_\omega b \text{ implies } a > b$$

They are key properties of our development, allowing us to switch back and forth between properties of $>$ and $>_\omega$, when convenient. They are easily proved in Coq.

R3. Special Properties of $>_\omega$ The two last properties to prove to fulfil the requirements of Bridges' axiom system are the following ones:

1. **Axiom of Archimedes**: For each $X \in \mathcal{HR}_\omega$ there exists a constructive $n \in A$ such that $X < n$.
2. **The constructive least-upper-bound principle** (see next section)

Archimedes property can be easily formalized in Coq:

```
Lemma Archimedes : forall X:HRw, exists n:HRw, n >=w X.
```

Proof. Its proof is immediate because the elements x of $\mathcal{HR}_\omega$ are such that there exists a limited $k \in \mathbb{N}$, $|x| < k\omega$. So the property can be proved using the integer $k\omega$ as a witness for n.

4.3 Least upper bound

To begin, let us recall some definitions. A subset S of $\mathcal{HR}_\omega$ is the collection of elements of $\mathcal{HR}_\omega$ which satisfies a given property defined in the system. This property may be internal or external. Such a subset S is *bounded above with respect to the relation* $\geqslant_\omega$ if there is $b \in \mathcal{HR}_\omega$ such that $b \geqslant_\omega s$ for all $s \in S$; the element b is called an *upper bound* of S. A *least upper bound* for S is an element $b \in \mathcal{HR}_\omega$ such that

- $\forall s \in S \quad b \geqslant_\omega s$ (b is an upper bound of S);
- $\forall b' \ (b >_\omega b') \Rightarrow (\exists s \in S \quad s >_\omega b')$.

A least upper bound is unique: if b and c are two least upper bounds of S, then we have $\neg(b >_\omega c)$ and $\neg(c >_\omega b)$; thus, according to the properties[8] of the relations $>_\omega$, $\geqslant_\omega$ and $=\omega$, we get $c \geqslant_\omega b$ and $b \geqslant_\omega c$ and then $b =_\omega c$.

The constructive least-upper-bound principle: Let S be a nonempty subset of $\mathcal{HR}_\omega$ that is bounded above w.r.t. the relation $\geqslant_\omega$, such that for all $\alpha, \beta \in \mathcal{HR}_\omega$ with $\beta >_\omega \alpha$, either β is an upper bound of S or else there exists $s \in S$ with $s >_\omega \alpha$; then S has a least upper bound.

Proof. To formalize and prove this property correct in Coq we follow the proof proposed in [8], which itself uses the heuristic motivation given by Bridges in [3]. It consists of two parts: we first construct a candidate b for the least upper bound and we then check that b is actually the least upper bound.

Definitions in Coq The subset property is defined as a property, i.e. $S\ x$ means x belongs to the set S. Then the notions of *least upper bound* and of *upper bound* are defined.

```
Definition subset := HRw->Prop.

Definition least_upper_bound (S:subset) (b:HRw) : Prop :=
 (forall s:HRw, (S s -> b >=w s)) /\
 (forall b':HRw, (b >w b') -> exists o:HRw, S o/\ o >w b').

Definition upper_bound (X:subset) (m:HRw) : Prop :=
 forall x:HRw, X x -> m >w x.
```

[8] These properties are not completely trivial in intuitionistic logic.

2 sequences (s_n, b_n) In the original paper [8], four sequences $(s_n, b_n, \alpha_n, \beta_n)$ which are mutually recursive and that will be useful to define a candidate b for the least upper bound are defined. To make it simpler in Coq, we have to simplify this definition by considering only 2 sequences (s_n, b_n). Indeed, one can easily note that α_n and β_n only depend on the values of s_n and b_n. Thus there is no need to include them in the recursive process of building the sequences (s_n) and (b_n). Indeed, at each step, α_n and β_n can be redefined as follows:

$$\alpha_n = \frac{2}{3} \times s_n + \frac{1}{3} \times b_n \qquad \beta_n = \frac{1}{3} \times s_n + \frac{2}{3} \times b_n.$$

Computing the next terms s_{n+1} and b_{n+1} of the sequences depends on the two preceding terms (s_n, b_n). From these data, we can build α_n and β_n as well as a proof of the property $\beta_n >_\omega \alpha_n$. However we need to carry around the three key properties of s_n and b_n:

- s_n belongs to S
- b_n is an upper bound of S
- $b_n -_\omega s_n =_\omega (\frac{2}{3})^n \times_\omega (b_0 -_\omega s_0)$.

Therefore, we choose to specify the function as precisely as possible when defining it, hence the numerous postconditions characterizing the output (s_n, b_n) of the function when giving the parameters n and Hn a proof that $\texttt{lim}\ n \wedge 0 \leqslant n$.

```
Definition def_s_b :
    forall n:A, forall Hn:(lim n /\ 0 <= n),
{sn:HRw & {bn:HRw &
   S sn /\
   upper_bound S bn /\
   bn +w (-w sn) =w ((power two_third n Hn)*w (b0 +w (-w s0)))}}.
```

The notation $\{x : A \& Q\ x\}$ stands for the (computational) existential quantification, i.e. there exists x of type A such that $Q\ x$ holds.

Initially, we have (s_0, b_0) with an arbitrary element s_0 of S, b_0 an upper bound of S, and $\alpha_0 = \frac{2}{3}s_0 + \frac{1}{3}b_0$ and $\beta_0 = \frac{1}{3}s_0 + \frac{2}{3}b_0$.

This requires the assumption that in Coq that we can always choose an arbitrary element s of S. This corresponds to a form of choice which can be expressed as follows:

```
Axiom non_empty : {x:HRw|X x}.
```

It states that a subset X of elements of $\mathcal{HR}_\omega$ is non-empty if and only if there exists an element x of $\mathcal{HR}_\omega$ for which X (x) holds.

Suppose for a given n, we have (s_n, b_n). We can build (α_n, β_n) as well as a proof of the property $\alpha_n <_\omega \beta_n$. By hypothesis, two different cases can happen:

- **First case** β_n is an upper bound of S. Then we choose s_{n+1} and b_{n+1} such that $s_{n+1} = s_n$ and $b_{n+1} = \beta_n$.
- **Second case** there exists s such that $(S\ s)$ and that $\alpha_n <_\omega s$. Then we choose s_{n+1} and b_{n+1} such that $s_{n+1} = s$ and $b_{n+1} = b_n + s - \alpha_n$.

Key properties. Several key properties of the elements of the sequences are already expressed in the type of `def_s_b`. They hold by construction (i.e. they are established using induction at the same time the actual sequences are computed). Among them, we know that, for each n, which is limited and positive, the property $S(s_n) \wedge$ `upper_bound` $S\ b_n$ holds. Thus, we can deduce that for any k and n, we have $b_k >_\omega s_n$. We also have the property that b_n and s_n are connected by the relation

$$b_n -_\omega s_n =_\omega (\frac{2}{3})^n \times_\omega (b_0 -_\omega s_0).$$

In addition to all the properties specified in the type of `def_s_b`, we also need to establish that the sequence (s_n) is increasing. Although this is immediate from its mathematical definition, it is rather challenging to prove it correct in Coq. Indeed it requires to apply the *ad hoc* induction principle `nat_like_induction` and its associated reduction rules without any assistance of the Coq proof engine. This happens to be fairly tedious in Coq because existential Σ-types and *let-in* constructs are involved in the definition of `def_s_b` and foldings of expanded definitions (e.g. the recursive call to `nat_like_induction`) have to be done manually [9].

Thanks to axiom ANS5, the sequences (s_n) and (b_n) can be extended to all integers, including non-limited ones. In addition, the overspill principle allows us to show the existence of an infinitely large number ν such that the following property holds:

$$\min_{0 \leqslant k \leqslant \nu} b_k \geqslant s_\nu \geqslant \ldots \geqslant s_1 \geqslant s_0.$$

A candidate for the least upper bound of S : $b := \min_{0 \leqslant k \leqslant \nu} b_k$

The next step, which consists of checking that b is the least upper bound is presented in [8] and is summarized here by proving the two following properties:

- on the one hand, that b is an upper bound of S,
- on the other hand, that b is actually a least upper bound, i.e. that for all $b' <_\omega b$, there exists $s \in S$ such that $s >_\omega b'$.

Proof scripts are available online (especially in the files `LUB*.v`) :

`http://dpt-info.u-strasbg.fr/~magaud/Harthong-Reeb/`

So far, we showed formally that the Harthong-Reeb line, constructed from a minimal axiom system for non standard arithmetic such as the one proposed in Section 3, does verify Bridges' axiomatic definition of what a constructive real line is.

[9] It requires *copy-pasting* the whole goal and using the `change` tactic to replace one term by the other, eventually making the system aware both terms are actually the same. Indeed, it actually agrees the 2 terms are convertible but can not perform any reductions leading from the first one to the expected one.

5 Instantiating the Harthong-Reeb line with Ω-numbers

In this section, we investigate how to provide some computational contents to the axiomatization of the Harthong-Reeb line. We implement non-standard integers using the Ω-numbers of Laugwitz and Schmieden. We first define these numbers and their operations and also prove their properties in Coq. We then discuss their limitations with respect to being a well-suited model for our axiom system for non-standard arithmetic and our formalization of the Harthong-Reeb line. Finally, we show that this implementation of non-standard numbers can be used to compute continuous functions using integers.

5.1 Formalization of the Ω-numbers

The Ω-numbers of Laugwitz and Schmieden permit the extension of a classical numerical system to a nonstandard one. Here we present this extension of integers. It can be viewed as a model of the axiomatic definition of the nonstandard arithmetic presented in Section 4. In their papers [20,21,22], Laugwitz and Schmieden extend the rational numbers and show that their system is equivalent to classical real numbers. In this section, we will not describe the whole theory but only introduce the basic notions that are essential to understand the Harthong-Reeb line. For more details about our approach please refer to [9].

To extend a theory of integer numbers, Laugwitz and Schmieden introduce a new symbol Ω to the classical ones $(0, 1, 2, 3, \ldots, +, /, \ldots)$. The only property that Ω verifies is named the *Basic Definition* (denoted (BD)) :

Definition 3. *Let $S(n)$ be a statement in $\mathbb{N}$ depending on $n \in \mathbb{N}$. If $S(n)$ is true for almost $n \in \mathbb{N}$, then $S(\Omega)$ is true.*

We consider the expression "*almost* $n \in \mathbb{N}$" means "for all $n \in \mathbb{N}$ from some level N", i.e. "$(\exists N \in \mathbb{N})$ such that $(\forall n \in \mathbb{N})$ with $n > N$". Since Ω can be substituted to any natural number, it denotes an Ω-number which is the first example of Ω-integer. Hence, each element a of this theory will be declined as a sequence $(a_n)_{n\in\mathbb{N}}$. Immediately, we can verify that Ω is infinitely large, i.e. greater than every element of $\mathbb{N}$. Indeed, for $p \in \mathbb{N}$, we apply (BD) to the statement $p < n$ which is true for almost $n \in \mathbb{N}$; thus $p < \Omega$ for each $p \in \mathbb{N}$. And Ω is the sequence $(n)_{n\in\mathbb{N}}$.

Technically speaking, Ω-integers are defined in Coq as sequences indexed by natural numbers (`nat`), whose values are relative integers (`Z`). The function `Z_of_nat` simply injects natural numbers into $\mathbb{Z}$.

```
Definition A := nat->Z.

Definition a0 : A := fun (n:nat) => 0%Z.
Definition a1: A := fun (n:nat) => 1%Z.

Definition w :A := fun (n:nat) => (Z_of_nat n).
```

Here we choose to define ω as the sequence $w_n = n$. However, we parametrized our development so that we can choose between three different ω, defined as $w_n = n$, $w_n = n^2$ or $w_n = 2^n$. The only requirement is that (w_n) must be strictly increasing. The sequence $w_n = 2^n$ is the most convenient one because all its terms are non-zero, which makes dealing with division easier in the implementation (see Section 5.3 for details).

To compare such Ω-numbers, we put the following equivalence relation:

Definition 4. *Let $a = (a_n)_{n\in\mathbb{N}}$ and $b = (b_n)_{n\in\mathbb{N}}$ be two Ω-numbers, a and b are equal if there exists $N \in \mathbb{N}$ such that for all $n > N$, $a_n = b_n$.*

This is captured by the definition `equalA`.

```
Definition equalA (u v:A) :=
 exists N:nat, forall n:nat, n>N -> u n=v n.
```

We define two classes of elements in this nonstandard theory:

- the class of *limited* elements: they are the elements $\alpha = (\alpha_n)_{n\in\mathbb{N}}$ which verify $\exists p \in \mathbb{Z}$ such that $\exists N \in \mathbb{N},\ \forall n > N, \alpha_n < p$ (example: $(2)_{n\in\mathbb{N}}$).
- the class of elements: *infinitely large numbers*, which are the sequences $\alpha = (\alpha_n)_{n\in\mathbb{N}}$ such that $lim_{n\to+\infty}\alpha_n = +\infty$

We recall that infinitely large numbers correspond to non-limited numbers.

The definition of the operations and relations on A, the set of Ω-numbers are the following ones. It is straightforward to formalize them in Coq.

Definition 5. *Let $a = (a_n)_{n\in\mathbb{N}}$ and $b = (b_n)_{n\in\mathbb{N}}$ two Ω-numbers,*

- $a + b =_{def} (a_n + b_n)_{n\in\mathbb{N}}$ *and* $-a =_{def} (-a_n)$ *and* $a \times b =_{def} (a_n \times b_n)_{n\in\mathbb{N}}$;

```
Definition plusA (u v:A) := fun (n:nat) => Zplus (u n) (v n).
Definition multA (u v:A) := fun (n:nat) => Zmult (u n) (v n).
Definition oppA (u:A) := fun (n:nat) => Zopp (u n).
```

- $a > b =_{def} [(\exists N \forall n > N)\ a_n > b_n]$ *and* $a \geqslant b =_{def} [(\exists N \forall n > N)\ a_n \geqslant b_n]$;

```
Definition leA (u v:A) :=
   exists N:nat, forall n:nat, n>N -> Zle (u n) (v n).

Definition ltA (u v:A) :=
   exists N:nat, forall n:nat, n>N -> Zlt (u n) (v n).
```

- $|a| =_{def} (|a_n|)$.

```
Definition absA (u:A) := fun (n:nat) => Zabs (u n).
```

Specificity of the theory Regarding the order relation, the usual properties true on $\mathbb{Z}$ are not always verified on $\mathbb{Z}_\Omega$ (= A, which is $\mathbb{N} \to \mathbb{Z}$). For instance

$$(\forall a, b \in \mathbb{Z}_\Omega) \quad (a \geqslant b) \vee (b \geqslant a) \tag{1}$$

is not valid as we can see for the particular Ω-integers $a = ((-1)^n)_{n\in\mathbb{N}}$ and $b = ((-1)^{n+1})_{n\in\mathbb{N}}$. Nevertheless, given two arbitrary Ω-numbers $a = (a_n)$ and $b = (b_n)$, we can prove

$$(\forall n \in \mathbb{N}) \quad (a_n \geqslant b_n) \vee (b_n \geqslant a_n) \tag{2}$$

because we have a decidability property for $\mathbb{Z}$. Using the basic definition (BD), we obtain $(a_\Omega \geqslant b_\Omega) \vee (b_\Omega \geqslant a_\Omega)$ and thus (1) since $a_\Omega = a$ and $b_\Omega = b$. Hence, there is a contradiction. To avoid it, we might admit that the application of (BD) leads to a notion of truth weaker than the usual notion.

Overall, this shows that the Ω-numbers can not be an exact model of the theory presented in Section 4. The main issue is the decidability property `A0_dec` of the order relation which is obviously not provable in this setting. In the next section, we show how the theory of Section 4 can be modified into a new (slightly different) theory of which the Ω-numbers are a model.

Nonstandard axioms We define two predicates `std` and `lim`: (`std` n) states that an Ω-number n is standard (i.e. denoted by a sequence which is constant from a given rank) and (`lim` n) that n is limited (i.e. its absolute value is bounded by a positive, standard number).

```
Definition std (u:A) :=
 exists N:nat, forall n m, n>N -> m>N -> (u n)=(u m).

Definition lim (a:A) :=
 exists p, std p /\ leA a0 p /\ ltA (absA a) p.
```

From these definitions, we can derive proofs of the axioms ANS1 to ANS4 presented in Section 4. In this paper, we did not investigate whether ANS5 can be defined constructively and it remains an open question to know whether ANS5 could be proved formally in Coq for Ω-numbers.

5.2 A version of the Harthong-Reeb line based on Ω-numbers

An instantiation of the minimal axiom system for non-standard arithmetic All properties of our minimal axiom system for non-standard arithmetic (presented in Section 3) need to be actually proved in the setting of Laugwitz-Schmieden Ω-numbers. It requires the design of some sophisticated tactics using Ltac, to deal with existential statements efficiently. Let us consider the following example (A is actually a functional type `nat`$\to$`Z`).

$$\texttt{lt_plus} : \forall x\, y\, z\, t : A, x < y \Rightarrow z < t \Rightarrow (x + z) < (y + t).$$

Proving this property, which corresponds (once everything is unfolded) to:

$$\begin{array}{l}\forall x\, y\, z\, t : \texttt{nat} \to \texttt{Z},\\ \forall H : \exists P : nat, \forall p : \texttt{nat}, p > P \to (x\ p < y\ p),\\ \forall H0 : \exists Q : \texttt{nat}, \forall q : \texttt{nat}, n > Q \to (z\ q < t\ q),\\ \quad \exists N : \texttt{nat}, \forall n : \texttt{nat}, n > N \to (x\ n + z\ n < y\ n + t\ n)\end{array}$$

requires extracting two witnesses P and Q, building a new witness $N := P+Q+1$ and generalizing the hypotheses coming from the destruction of H (resp. $H0$) with the universally quantified variable n and a proof that it is greater than P (resp. Q) - which is the case as it is greater than P+Q+1 by hypothesis - . This yields an inequation on `Z`, which can be solved using `omega` or the adequate lemma of the `ZArith` library. The tactics `solve_ls` and `solve_ls_w_l` (which provides a lemma of `ZArith` as a hint) perform all this in one go.

Limitations of the Laugwitz-Schmieden integers Laugwitz-Schmieden integers cannot be a model of the Harthong-Reeb line as it was defined in Section 4. In [7], some of the authors identified among all the properties required to be a Bridges-Heyting field the ones which are not immediately verified by the instantiation of the Harthong-Reeb line with Ω-numbers. This is summarized in a the theorem (from [7]) which is quoted in Figure 3. It expresses that three of Bridges axioms do not hold (namely A.2.2, A2.3 and A3.2) when non-standard integers are represented as Laugwitz-Schmieden integers. However, these three statements can be (slightly) modified so that they hold on Laugwitz-Schmieden integers (see [7] for details).

Theorem 1. *The system $\mathcal{HR}_\omega$[10] has the following properties:*
(a) *All the axioms of a BH-ordered field except* BH2.(2), BH2.(3) *and* BH3.(2).
(b) BH2.(2)' *If $x, y \in \mathcal{HR}_\omega$ are such that $x <_\omega y$, then for each $z \in \mathcal{HR}_\omega$, there is a $q \in \mathbb{N}$ such that $(q(y-z) \geqslant \omega) \vee (q(z-x) \geqslant \omega)$ is weakly true.*
(c) BH2.(3)' *If $x, y \in \mathcal{HR}_\omega$ are such that $x \triangle y$ and $\neg(x \neq_\omega y)$, then $x =_\omega y$.*
(d) BH3.(2)' *If S is a nonempty subset of $\mathcal{HR}_\omega$ that is bounded above relative to the relation $\geqslant_\omega$ and such that for all $(\alpha, \beta) \in \mathcal{HR}_\omega^2$ where β is an upper bound of S and $\alpha \in S$ and for $(a, b) \in \mathcal{HR}_\omega^2$ such that $\alpha \leqslant_\omega a \leqslant_\omega b \leqslant_\omega \beta$, either b is an upper bound of S or else there exists $s \in S$ with $s >_\omega a$.*
Then there exists an element $\tau \in \mathcal{HR}_\omega$ which is a least upper bound of S in the following weak meaning:
(I') *$\forall \mu <_\omega \tau$, $\exists s \in S$ such that $\mu <_\omega s$ (identical to* (I)*)*
(II') *$(\forall \nu \in \mathcal{HR}_\omega$ such that $\tau <_\omega \nu)(\exists b$ upper bound of $S)$ $\tau \leqslant_\omega b <_\omega \nu$*

Fig. 3. The properties of $\mathcal{HR}_\omega$ based on Laugwitz-Schmieden Ω-numbers (from [7])

Our formalization in Coq confirms these issues. Let us remind the reader the problem comes with the axiom (A is instantiated with `nat`→`Z`):

```
Axiom A0_dec : forall x: A, {x < 0}+{x = 0}+{0 < x}.
```

Indeed, as stated in the previous section, we have no decidability of the order for sequences a and b in general, we can only prove a decidability property for each of the elements of the sequences (a_n) and (b_n).

Axioms A2.2, A2.3 and A3.2 (the least upper bound principle)[11] can not be proved in this setting as stated in [6,7]. In addition, basic properties such as generalizing the property that a sequence is increasing interfere with these decidability properties and their proofs as they were carried out in Section 4 are no longer acceptable. This is the case for the following statement:

$$\forall n, s_n \leq s_{n+1} \Rightarrow (\forall p\, q, p \leq q \Rightarrow s_p \leq s_q).$$

Formalizing the theorem quoted in Figure 3 is currently under way in our formal development. It requires us to take into account the specificities of Laugwitz-Schmieden numbers compared to our first axiom system for non-standard arithmetic. We hope that proofs presented in [7] could be easily formalized in Coq.

5.3 Computing continuous functions

The main advantage of having a concrete representation of non-standard integers is the possibility to compute with them. This is what we achieve in this section using the Ω-integers. It provides us with practical means to compute continuous functions using only integer sequences.

This is based on the Ω-arithmetization scheme as described in [7]. An arithmetization scheme is a process which provides discrete equivalents to continuous functions. The one we use is derived from the well known Euler's integration scheme

$$\begin{cases} T_0 = A; X_0 = B \\ T_{k+1} = T_k + \frac{1}{h} \\ X_{k+1} = X_k + \frac{1}{h} \times F(T_k, X_k) \end{cases}$$

This scheme computes a numerical approximation $(T_k, X_k)_{k \geqslant 0}$ of the continuous function $X : T \mapsto X(T)$ defined on an interval of $\mathbb{R}$ with values in $\mathbb{R}$ which is the solution of the Cauchy problem $X^{'} = F(X, T),\ X(A) = B$. It is well known that the error of the approximation $|X(T_k) - X_k|$ converges to 0 when $h \to +\infty$.

Following Reeb and Reveillès [31], the idea is to replace the number h with an infinitely large Ω-integer and to translate all other quantities using the function $\Psi_\omega : \mathbb{R} \to \mathbb{Z}_\Omega$ such that $\Psi_\omega(U) = (\lfloor \omega_m \times U \rfloor)_{m \in \mathbb{N}}$ and where ω is an infinitely large Ω-integer.

The Ω-arithmetization of Euler's scheme at the global scale ω is the following scheme with variables $x_k, t_k \in \mathbb{Z}_\Omega$ (i.e. sequences of elements of $\mathbb{Z}$) :

$$\begin{cases} t_0 = a; x_0 = b \\ t_{k+1} = t_k + \beta \\ x_{k+1} = x_k + f(t_k, x_k) \div \beta \end{cases}$$

[11] They correspond to axioms BH2.(2), BH2.(3), BH3.(2) in Fig. 3.

where β is an infinitely large positive Ω-integer such that $\beta^2 = \omega$ and $a = \Psi_\omega(A)$, $b = \Psi_\omega(B)$ and f is the translation of the function F,

$$f(t,x) = (\lfloor \omega_m \times F(t_m/\omega_m, x_m/\omega_m) \rfloor)_{m\in\mathbb{N}}.$$

This scheme computes a sequence of points (t_k, x_k) which is the graph of a discrete function $t \mapsto x(t)$ called the Ω-arithmetization at the global scale ω of the function $T \mapsto X(T)$. The problem with this direct translation is that the domain of the function $t \mapsto x(t)$ is not connected since $t_{k+1} - t_k = \beta$ which is infinitely large. To correct this, the arithmetic scaling $x \mapsto x \div \beta$ is used. This sends β to 1 and now the points (t_k, x_k) are observed at the intermediate scale β. In order to compute the arithmetization directly at the scale β, the following scheme is used

$$\begin{cases} \widetilde{t}_0 = a \div \beta, \widetilde{x}_0 = b \div \beta \text{ and } \widehat{x}_0 = b \bmod \beta \\ \widetilde{t}_{k+1} = \widetilde{t}_k + 1 \\ \widetilde{x}_{k+1} = \widetilde{x}_k + (\widehat{x}_k + \widetilde{f}_k) \div \beta \\ \widehat{x}_{k+1} = (\widehat{x}_k + \widetilde{f}_k) \bmod \beta \end{cases}$$

where $\widetilde{x} = x \div \beta$ and $\widehat{x} = x \bmod \beta$ and $\widetilde{f}_k = f(\widetilde{t}_k \times \beta + a \bmod \beta, \widetilde{x}_k \times \beta + \widehat{x}_k) \div \beta$. This scheme computes a sequence of points $(\widetilde{t}_k, \widetilde{x}_k)$ which is the graph of the discrete function $\widetilde{t} \mapsto \widetilde{x}(\widetilde{t})$ defined over a connected domain because $\widetilde{t}_{k+1} - \widetilde{t}_k = 1$. This discrete function is the Ω-arithmetization of the function $T \mapsto X(T)$ at the intermediate scale β.

Properties of Ω-arithmetization are studied in [7]. The most notable is that Ω-arithmetization is an exact representation of the continuous function $T \mapsto X(T)$. So, this is a good motivation to experiment computations using the formalization of the Harthong-Reeb line with Coq.

The previous scheme is straightforward to implement in Coq using recursive functions:

```
Fixpoint LS_Euler_stepA
  beta a b (f:A *  A -> A) m : list (A * A * A) :=
  match m with
  | O => (a / beta, b / beta, b mod% beta)::nil
  | (S k) =>
    let r := (LS_Euler_stepA beta a b f k) in
    let '(ttk, xtk, xhk) := hd (a0, a0, a0)  r in
    let ftk :=
      (f ((ttk * beta) + (a mod% beta) , (xtk * beta) + xhk)) / beta
    in
    ( ttk + a1,
     xtk + (xhk + ftk) / beta,
     (xhk + ftk) mod% beta ) :: r
end.
```

Note that the function `hd` that returns the head of a list comes with an extra argument in case it receives the empty list as argument.

The computational contents of our formal description of the Ω-integers of Laugwitz and Schmieden, including the function computing the arithmetization is automatically extracted into the functional programming language Ocaml. The main extraction commands required to do that are summarized in Figure 5. The extracted code can then be connected to a visualisation tool which allows us to observe the continuous function we computed with the arithmetisation scheme.

Extraction consists in removing all terms with non computational contents and making all dependent types non-dependent such that they can be accepted by Ocaml's type-checker. The commands `Extract Inductive` and `Extract Constant` provide means to explain to the system how Coq inductive definitions and constants should be implemented in Ocaml. There is no formal verification of the relevance of this translation. It is the user's responsability to make sure no mistakes are introduced at this stage. Recent improvements of the Coq computation machinery shall make it possible to compute our continuous functions directly in Coq, thus avoiding this (unsafe) extraction process to Ocaml.

For example, the result of the arithmetization for the function $t \mapsto \frac{t^2}{6}$ is shown in Figure 4. In this example, the function corresponding to F in the above algorithm is $F : (T, X) \mapsto \frac{T}{3}$. As Ω-integers are infinite sequences of integers they cannot be visualized directly. One must choose different fixed ranks in the involved sequences to see the graphs of the discrete function $\widetilde{t} \mapsto \widetilde{x}(\widetilde{t})$ at those ranks. Examples of graphs at different ranks for the function $t \mapsto \frac{t^2}{6}$ are drawn on the Figure 4.

6 Discussion

The actual construction of $\mathcal{HR}_\omega$ based on Laugwitz-Schmieden so-called Ω-integers, proposed in Section 5.2 leads to an alternative constructive continuum which is different from the one characterized by Bridges' axiom system. Formalizing it is another challenge and is currently under way. In addition to shifting the reasoning from the level of the sequences to the level of the elements (of `Z`), it also requires formalizing several new notions, especially a congruence relation on sequences which allows us to define what *regular* Ω-integers are. From what we experimented and reported so far, and even if we are confident the proofs of [7] are correct, it seems very useful to carry out this formalization. It would allow for a better understanding of the definitions, properties and associated proofs and may lead to fixing some imprecisions or shortcomings of the hand-written proofs.

Future work also include linking our formal description of the Harthong-Reeb line with a constructive description of real numbers such as the one developed in Niemejen [30,19]. This would require to define Ω-rationals and then show formally that *the Harthong-Reeb system is equivalent to* $\mathbb{R}$ *in some sense* (the exact meaning of this sentence is discussed in [8]). Once this is achieved, we could formally prove that the Euler arithmetization scheme, whose implementation in Coq was presented in this paper, is actually correct.

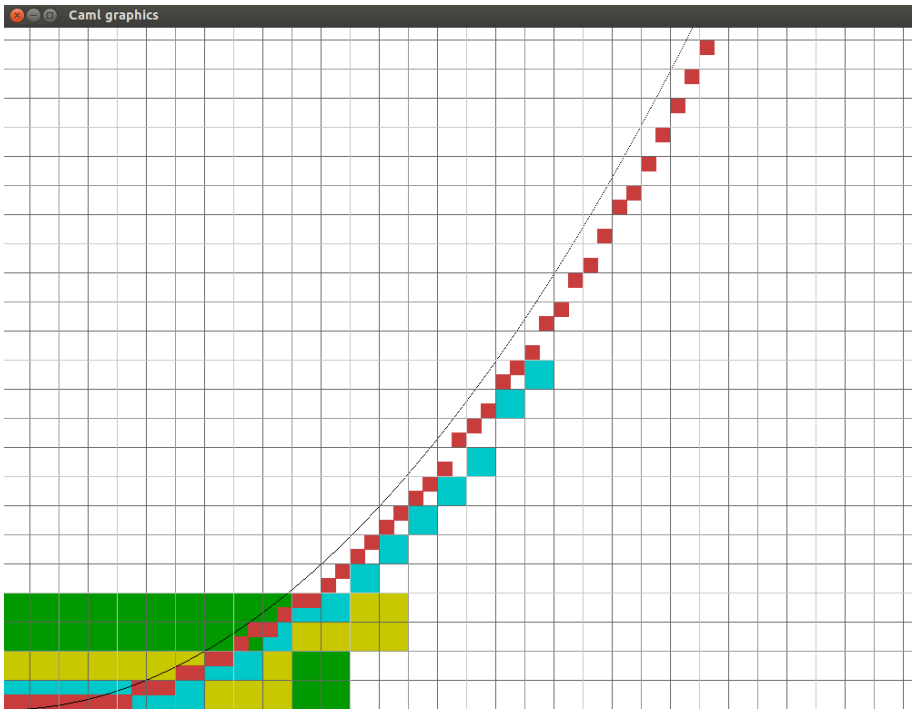


Fig. 4. The arithmetization of the function $t \mapsto \frac{t^2}{6}$. Graphs of the function $\widetilde{t} \mapsto \widetilde{x}(\widetilde{t})$ are drawn at different ranks of the given infinite integer sequences.

7 Conclusion

We proposed a formalization of non standard arithmetic on top of which we built the Harthong-Reeb line. We also implemented non standard arithmetic using Laugwitz-Schmieden Ω-numbers.

This includes proving that our formal description of the Harthong-Reeb line verifies Bridges' axioms, and therefore is a constructive real line. In addition, we verify that the Ω-numbers are a suitable implementation of non standard arithmetic, with which all these properties can be proved correct, with the exceptions of R2.2. and R2.3. and the least upper bound property. Figure 6 provides an insight of the size of the development. All proofs are available online :

`http://dpt-info.u-strasbg.fr/~magaud/Harthong-Reeb/`

We also managed to extract the Coq definitions into an Ocaml program which performs exact computations of continuous functions using only Laugwitz-Schmieden (sequences of) integers.

As shown in [7], the axiom system for the Harthong-Reeb line can be adapted to make Laugwitz-Schmieden an adequate model of this theory. This still needs

```
Extraction Language Ocaml.

Extract Inductive prod => "(*)"  [ "(,)" ].
Extract Inductive bool => "bool" ["true" "false"].
Extract Inductive sumbool => "bool" ["true" "false"].

Extract Inductive Z => "int" ["0" "(fun x -> x)" "(fun x -> -x)"].
Extract Inductive positive => "int" ["(fun x -> 2*x+1)" "(fun x -> 2*x)" "1"].
Extract Inductive list => "list" ["([])" "(::)"].
Extract Constant Z_of_nat =>
  "(fun x -> let rec n2b x = match x with 0 -> 0 | S n -> (n2b n)+1 in n2b x)".

Extract Constant Zcompare =>
  "(fun x y -> if (x=y) then Eq else if (x<y) then Lt else Gt)".
Extract Constant Psucc => "(fun x -> x+1)".
Extract Constant Pplus => "(fun x y -> x+y)".
Extract Constant Pmult => "(fun x y -> x*y)".
Extract Constant Pminus => "(fun x y -> x-y)".
Extract Constant Pdouble_minus_one => "(fun x -> 2*x+1)".
Extract Constant Zdiv_eucl_POS => "(fun x y -> (x/y, x mod y))".
Extract Constant Zdiv_eucl => "(fun x y -> (x/y, x mod y))".

Extract Constant Zplus => "(fun x y -> x+y)".
Extract Constant Zmult => "(fun x y -> x*y)".
Extract Constant Zopp => "(fun x -> -x)".

Extract Constant Zge_bool => "(fun x y -> x >= y)".
Extract Constant Zgt_bool => "(fun x y -> x > y)".

Extract Constant A0_dec_weak_gen => "(fun x n ->
if (x n<0) then (Inleft true) else
if (x n=0) then (Inleft false) else (Inright))".

Extract Constant Zabs => "(fun x -> if x < 0 then -x else x)".
```

Fig. 5. A simplified version of the extraction commands

to be verified in Coq. Another research direction would be to combine Laugwitz-Schmieden integers and geometric algebras to perform computations of geometric predicates, e.g. the orientation predicate in the plane.

References

1. Y. Bertot and P. Castéran. *Interactive Theorem Proving and Program Development, Coq'Art: The Calculus of Inductive Constructions*. Springer, 2004.
2. D. Bridges and E. Palmgren. *Constructive Mathematics*. The Stanford Encyclopedia of Philosophy. 2013. available from `http://plato.stanford.edu/archives/win2013/entries/mathematics-constructive/`.

	specifications	proofs
Nonstandard arithmetic	110	20
$\mathcal{HR}_\omega$ and Bridges' axioms (without L.U.B.)	320	1700
(+ the least upper bound principle)	210	700
Laugwitz-Schmieden	220	580

Fig. 6. Key figures of our formal development (as computed by `coqwc`)

3. D. Bridges and S. Reeves. Constructive mathematics, in theory and programming practice. Technical Report CDMTCS-068, Centre for Discrete Mathematics and Theorical Computer Science, November 1997.
4. D. S. Bridges. Constructive mathematics: A foundation for computable analysis. *Theor. Comput. Sci.*, 219(1-2):95–109, 1999.
5. A. Chlipala. *Certified Programming with Dependent Types: A Pragmatic Introduction to the Coq Proof Assistant*. MIT Press, 2013.
6. A. Chollet. *Formalismes non classiques pour le traitement informatique de la topologie et de la géométrie discrète*. PhD thesis, Université de la Rochelle, 2010.
7. A. Chollet, G. Wallet, L. Fuchs, E. Andres, and G. Largeteau-Skapin. Foundational Aspects of Multiscale Digitization. *Theor. Comput. Sci.*, 466:2–19, 2012.
8. A. Chollet, G. Wallet, L. Fuchs, G. Largeteau-Skapin, and E. Andres. Insight in discrete geometry and computational content of a discrete model of the continuum. *Pattern recognition*, 42:2220–2228, 2009.
9. A. Chollet, G. Wallet, L. Fuchs, G. Largeteau-Skapin, and E. Andres. Ω-Arithmetization: a Discrete Multi-resolution Representation of Real Functions. In P. Wiederhold and P. R. Barneva, editors, *Combinatorial Image Analysis: 13th International Workshop, IWCIA 2009*, volume 5852 of *Lecture Notes in Computer Science (LNCS)*, pages 316–329, Mexico, November 2009.
10. Coq development team. *The Coq Proof Assistant Reference Manual, Version 8.2*. LogiCal Project, 2008.
11. F. Diener and G. Reeb. *Analyse Non Standard*. Hermann, Paris, 1989.
12. M. Diener. Application du calcul de Harthong-Reeb aux routines graphiques. In J.-M. Salanskis and H. Sinaceurs, editors, *Le Labyrinthe du Continu*, pages 424–435. Springer, 1992.
13. J. Fleuriot. Exploring the foundations of discrete analytical geometry in Isabelle/HOL. In P. Schreck, J. Richter-Gebert, and J. Narboux, editors, *Proceedings of Automated Deduction in Geometry 2010*, volume 6877 of *LNAI*. Springer, 2011.
14. J. D. Fleuriot and L. C. Paulson. A combination of nonstandard analysis and geometry theorem proving, with application to newton's principia. In C. Kirchner and H. Kirchner, editors, *CADE*, volume 1421 of *Lecture Notes in Computer Science*, pages 3–16. Springer, 1998.
15. H. Geuvers, M. Niqui, B. Spitters, and F. Wiedijk. Constructive analysis, types and exact real numbers. *Mathematical Structures in Computer Science*, 17(1):3–36, Mar. 2007.
16. J. Harthong. Une théorie du continu. In H. Barreau and J. Harthong, editors, *La mathématiques non standard*, pages 307–329, Paris, 1989. Éditions du CNRS.
17. M. Huth and M. Ryan. *Logic in Computer Science*. Cambridge University Press, 2nd edition edition, 2004.
18. R. Kaye. *Models of Peano Arithmetic*. Oxford Science Publications, 1991.

19. R. Krebbers and B. Spitters. Type Classes for Efficient Exact Real Arithmetic in Coq. *Logical Methods in Computer Science*, 9(1), 2011.
20. D. Laugwitz. Ω-calculus as a generalization of field extension an alternative approach to nonstandard analysis. In A. Hurd, editor, *Nonstandard Analysis - Recent developments*, volume 983 of *Lecture Notes in Mathematics*, pages 120–133. Springer, 1983.
21. D. Laugwitz. Leibniz' principle and Ω-calculus. In J. Salanskis and H. Sinacoeur, editors, *Le Labyrinthe du Continu*, pages 144–155. Springer France, 1992.
22. D. Laugwitz and C. Schmieden. Eine erweiterung der infinitesimalrechnung. *Mathematische Zeitschrift*, 89:1–39, 1958.
23. R. Lutz. La force modélisatrice des théories infinitésimales faibles. In J.-M. Salanskis and H. Sinaceur, editors, *Le Labyrinthe du Continu*, pages 414–423. Springer-Verlag, 1992.
24. N. Magaud, A. Chollet, and L. Fuchs. Formalizing a Discrete Model of the Continuum in Coq from a Discrete Geometry Perspective. In *ADG'2010*, 2010. Accepted for presentation at the conference.
25. P. Martin-Löf. *Intuitionnistic Type Theory.* Bibliopolis, Napoli, 1984.
26. P. Martin-Löf. Mathematics of infinity. In *COLOG-88 Computer Logic*, Lecture Notes in Computer Science, pages 146–197. Springer-Verlag Berlin, 1990.
27. I. Moerdijk. A model for intuitionistic non-standard arithmetic. *Ann. Pure Appl. Logic*, 73(1):37–51, 1995.
28. E. Nelson. Internal set theory: A new approach to nonstandard analysis. *Bulletin of the American Mathematical Society*, 83(6):1165–1198, November 1977.
29. E. Nelson. *Radically Elementary Theory.* Annals of Mathematics Studies. Princeton University Press, 1987.
30. R. O'Connor. Certified Exact Transcendental Real Number Computation in Coq. In O. A. Mohamed, C. A. Muñoz, and S. Tahar, editors, *TPHOLs*, volume 5170 of *Lecture Notes in Computer Science*, pages 246–261. Springer, 2008.
31. J.-P. Reveillès and D. Richard. Back and forth between continuous and discrete for the working computer scientist. *Annals of Mathematics and Artificial Intelligence, Mathematics and Informatic*, 16(1-4):89–152, 1996.