New Interactions between Analysis, Topology, and Computation
Birmingham, 7 January 2009

A coinductive approach to exact real number computation

Ulrich Berger
Swansea

The aims of this talk

- to outline a constructive theory of digital computation based on coinduction, and its application to computable analysis
- to show that program extraction from proofs is a practical method for obtaining certified programs

Outline

- Introduction
- Induction and coinduction
- Digit spaces
- Program extraction
- Metric digit spaces
- Case studies
- Conclusion

Background

- **Exact Real Arithmetic via infinite streams of digits**
  Edalat, Potts, Sünderhauf, Heckmann, Escardó, Vuillemin, Ciaffaglione, Gianantonio, Bertot, Niqui, Blanck ...

- **Coalgebraic modelling of infinite data**
  Jacobs, Rutten, Adamek, Kurz, Altenkirch, Ghani, Hancock, Pattinson, Escardó, Pavlovic, Pratt, ...

- **Program extraction from constructive proofs**
  Tatsuta, Schwichtenberg, Letouzey, O'Connor, Spitters, Bertot, Niqui, Bauer, Taylor, B, ...

- **Domain-theoretic modelling and termination proofs**
  Plotkin, Edalat, Pattinson, Escardó, Coquand, Spiwack, Buchholz, B ...

Example: computing with signed digits

$$\mathbb{I} := [-1, 1] \subseteq \mathbb{R} \qquad x \in \mathbb{I}$$
$$\mathrm{SD} := \{-1, 0, 1\} \qquad a = (a_n)_{n \in \mathbb{N}} \in \mathrm{SD}^\omega$$

$$x \sim a \quad :\Leftrightarrow \quad x = \sum_{n=0}^{\infty} a_n \cdot 2^{-(n+1)}$$

A function $f : \mathbb{I} \to \mathbb{I}$ is *represented* by a function $\hat{f} : \mathrm{SD}^\omega \to \mathrm{SD}^\omega$ if

$$\forall x, a \; (\, x \sim a \Rightarrow f(x) \sim \hat{f}(a) \,)$$

Power series as infinite composition

$$\sum_{n=0}^{\infty} a_n \cdot 2^{-(n+1)} = \frac{1}{2}(a_0 + \frac{1}{2}(a_1 + \ldots)) = \mathrm{av}_{a_0}(\mathrm{av}_{a_1}(\ldots))$$

where $\mathrm{av}_d : \mathbb{I} \to \mathbb{I}, \quad \mathrm{av}_d(x) := \frac{1}{2}(d + x) \quad (d \in \mathrm{SD})$.

Therefore

$$x \sim a \;\Leftrightarrow\; x = \mathrm{av}_{a_0} \circ \mathrm{av}_{a_1} \circ \ldots$$

$\mathrm{AV} := \{\mathrm{av}_{-1}, \mathrm{av}_0, \mathrm{av}_1\} \subseteq \mathbb{I} \to \mathbb{I}$.

$(\mathbb{I}, \mathrm{AV})$ is an example of a *digit space*.

Digit spaces

We study digit spaces (X, D), where X is a set and $D \subseteq X \to X$, and characterise the functions $f : X \to Y$ that have a continuous digital representation $\hat{f} : D^\omega \to E^\omega$.

The characterisation does not refer to infinite objects (like streams of digits), but uses a combined inductive/coinductive definition.

Program extraction yields implementations of $\hat{f}$ by finitely branching non-wellfounded trees.

We also consider *metric digit spaces* (X, σ, P, D), where σ is a metric on X and $P \subseteq X$ is dense, and study the relation between digital representability and uniform continuity.

Induction

$\Phi \colon \mathcal{P}(U) \to \mathcal{P}(U)$ is monotone if $X \subseteq Y$ implies $\Phi(X) \subseteq \Phi(Y)$.

A set $X \subseteq U$ is Φ-*closed* if $\Phi(X) \subseteq X$.

$\mu\Phi$, the set *inductively* defined by Φ, is the least Φ-closed set.

Closure	$\Phi(\mu\Phi) \subseteq \mu\Phi$
Induction	if $\Phi(X) \subseteq X$, then $\mu\Phi \subseteq X$

Coinduction

A set $X \subseteq U$ is Φ-*coclosed* if $X \subseteq \Phi(X)$.

$\nu\Phi$, the set *coinductively* defined by Φ, is the largest Φ-coclosed set.

Coclosure	$\nu\Phi \subseteq \Phi(\nu\Phi)$
Coinduction	if $X \subseteq \Phi(X)$, then $X \subseteq \nu\Phi$

Digital maps (motivation)

Let $x = \mathrm{av}_{a_0} \circ \mathrm{av}_{a_1} \circ \ldots$. Consider $h \colon \mathbb{I} \to \mathbb{I}$.

Writing a digit

$$\begin{aligned} h(x) &= h \circ \mathrm{av}_{a_0} \circ \mathrm{av}_{a_1} \circ \ldots \\ &= \mathrm{av}_b \circ ((\mathrm{av}_b^{-1} \circ h) \circ \mathrm{av}_{a_0} \circ \mathrm{av}_{a_1} \circ \ldots) \qquad \text{if } h[\mathbb{I}] \subseteq \mathrm{av}_b[\mathbb{I}] \end{aligned}$$

Reading a digit

$$\begin{aligned} h(x) &= h \circ \mathrm{av}_{a_0} \circ \mathrm{av}_{a_1} \circ \ldots \\ &= (h \circ \mathrm{av}_{a_0}) \circ \mathrm{av}_{a_1} \circ \ldots \end{aligned}$$

After reading finitely many digits writing must be possible again.

$\Rightarrow$ Edalat, Potts, Escardó, ...

Digital maps (informal definition)

Let (X, D) and (Y, E) be digit spaces.

For any set $F \subseteq X \to Y$ we define (inductively) $\mathcal{J}(F)$ as the least subset of $X \to Y$ such that

(W) if $e \in E$ and $f \in F$, then $e \circ f \in \mathcal{J}(F)$;
(R) if $f \circ d \in \mathcal{J}(F)$ for all $d \in D$, then $f \in \mathcal{J}(F)$.

The set C of *digital maps* is (coinductively) defined as the largest subset of $X \to Y$ such that

$$\mathrm{C} \subseteq \mathcal{J}(\mathrm{C})$$

i.e. C is the largest fixed point of $\mathcal{J}$.

Digital maps (formal definition)

Let (X, D) and (Y, E) be digit spaces.

We define the set $\mathrm{C} \subseteq X \to Y$ of digital maps as follows.

Let F, G range over subsets of $X \to Y$
and let $\nu F. \ldots$ stand for $\nu \lambda F \ldots$ e.t.c.

$$\begin{aligned} \mathrm{C} \quad := \quad & \nu F.\mu G.\{e \circ f \mid e \in E, f \in F\} \cup \\ & \qquad \{h : X \to Y \mid \forall d \in D\; h \circ d \in G\} \end{aligned}$$

When we wish to make explicit the dependency of C on the digit spaces (X, D) and (Y, E), we write $\mathrm{C}_{D,E}$ or even $\mathrm{C}_{(X,D),(Y,E)}$ for C.

Identity and composition

Identity Lemma

Let (X, D) be a digit spaces.

(a) $\mathrm{id}_X \in \mathrm{C}_{D,D}$.

(b) $D \subseteq \mathrm{C}_{D,D}$.

Composition Lemma

Let (X_i, D_i) (i=1,2,3) be digit spaces.

If $f \in \mathrm{C}_{D_1,D_2}$ and $g \in \mathrm{C}_{D_2,D_3}$, then $g \circ f \in \mathrm{C}_{D_1,D_3}$.

The category of digit spaces

By the Identity Lemma and the Composition Lemma, digit spaces and digital maps form a category.

Product Lemma

The category $\mathcal{D}$ has finite products.

Digital global elements

The set of global elements of a digit space (X, D) is

$$\mathrm{C}_D := \mathrm{C}_{\mathbf{1},(X,D)}$$

where $\mathbf{1}$ denotes the terminal object $(\mathbf{1}, \{\mathrm{id}_{\mathbf{1}}\})$ in $\mathcal{D}$. We identify C_D with a subset of X.

Global Element Lemma

$$\mathrm{C}_D = \nu A.\{d(x) \mid d \in D, x \in A\} \overset{\text{roughly}}{=} \{d_0 \circ d_1 \circ \ldots \mid (d_n)_{n \in \mathbb{N}} \in D^\omega\}$$

Application Lemma

If $f \in \mathrm{C}_{D,E}$ and $x \in \mathrm{C}_D$, then $f(x) \in \mathrm{C}_E$.

Proof: Composition Lemma.

Formalisation

- First-order logic with equality and free predicate constants and variables.
- Least and greatest fixed points $\mu X.\{\vec{x} \mid A\}$, $\nu X.\{\vec{x} \mid A\}$, where A is strictly positive in X.
- Axiom schemes for least and greatest fixed points (induction and coinduction are unrestricted).
- Other (mathematical) axioms must be *non-computational*, i.e. contain neither free predicate variables nor the propositional connective $\vee$.
- Intuitionistic logic.

General goal: The system should be support a *direct* formalisation of common mathematical theories.

Example

- Predicate constant ϵ for set membership.
- Ad-hoc individual constants: $\emptyset$, $\mathbb{R}$, $<$, 0, 1, $\ldots$.
- Ad-hoc function constants: $+$, $*$, $\ldots$.
- Non-computational axioms of set-theory.
- Non-computational axioms stating that $\mathbb{R}$ (more precisely $\{x \mid x\epsilon\mathbb{R}\}$) with $0, 1, +, *, <$ is a real closed field.

Example (ctd.)

Positive binary natural numbers:

$$\mu X.\{y \mid y = 1 \lor \exists x\, \exists c\, (X(x) \land c \in \{0,1\} \land y = 2 * x + c)\}$$

The real interval $\mathbb{I} = [-1, 1]$:

$$\begin{aligned} & \nu X.\{y \mid |y| \leq 1 \land \exists x\, \exists c\, (X(x) \land c \in \{0,1,-1\} \land y = \frac{x+c}{2})\} \\ \simeq\ & \mathrm{C}_{\mathrm{AV}} \end{aligned}$$

The general theory of digit spaces can be formalised as well.

Realisability

- Realisability language: Extension of the object language by a new sort for program terms.
- Program terms: untyped λ-terms with pairing/projections, injections/case analysis, and recursion.
- Equations for program terms as extra axioms.
- For each formula A in the object-language we define a predicate $\mathbf{r}(A)$ in the realisability language. The formula $\mathbf{r}(A)(M)$ (also written $M\,\mathbf{r}\,A$) intuitively means that the program term M "realises" A.

Definition of realisability

If A is non-computational:

$$\mathbf{r}(A) \quad = \quad \{() \mid A\}$$

If A is non-computational, but B is:

$$\begin{aligned} \mathbf{r}(A \wedge B) = \mathbf{r}(B \wedge A) \quad &= \quad \{x \mid A \wedge \mathbf{r}(B)(x)\} \\ \mathbf{r}(A \rightarrow B) \quad &= \quad \{x \mid A \rightarrow \mathbf{r}(B)(x)\} \end{aligned}$$

Definition of realisability (ctd.)

In all other cases:

$$
\begin{array}{rcl}
\mathbf{r}(X(\vec{t})) & = & \{x \mid \tilde{X}(x,\vec{t})\} \\
\mathbf{r}(A \wedge B) & = & \{\langle x,y\rangle \mid \mathbf{r}(A)(x) \wedge \mathbf{r}(B)(y)\} \\
\mathbf{r}(A \vee B) & = & \{\mathrm{inl}(x) \mid \mathbf{r}(A)(x)\} \cup \{\mathrm{inr}(y) \mid \mathbf{r}(B)(y)\} \\
\mathbf{r}(A \rightarrow B) & = & \{f \mid \forall x\,(\mathbf{r}(A)(x) \rightarrow \mathbf{r}(B)(fx))\} \\
\mathbf{r}(\forall y\, A) & = & \{x \mid \forall y\,(\mathbf{r}(A)(x))\} \\
\mathbf{r}(\exists y\, A) & = & \{x \mid \exists y\,(\mathbf{r}(A)(x))\} \\
\mathbf{r}(\mu X.\mathcal{P}) & = & \mu\tilde{X}.\{(x,\vec{y}) \mid \mathbf{r}(\mathcal{P}(\vec{y}))(x)\} \\
\mathbf{r}(\nu X.\mathcal{P}) & = & \nu\tilde{X}.\{(x,\vec{y}) \mid \mathbf{r}(\mathcal{P}(\vec{y}))(x)\}
\end{array}
$$

Soundness

From a closed derivation of a formula A one can extract a program term M and a derivation of $\mathbf{r}(A)(M)$.

Proof: Induction and coinduction are realised by terms modelling structural recursion and corecursion:

$$
\begin{aligned}
\mathbf{It}_{X,\mathcal{P}} s &= s \circ \mathbf{map}_{X,\mathcal{P}}(\mathbf{It}_{X,\mathcal{P}} s) \\
\mathbf{Coit}_{X,\mathcal{P}} s &= \mathbf{map}_{X,\mathcal{P}}(\mathbf{Coit}_X \mathcal{P} s) \circ s
\end{aligned}
$$

The terms $\mathbf{map}_{X,\mathcal{P}}$, which are defined simultaneously, realise the monotonicity of $\mathcal{P}$ w.r.t. to X.
In fact, the following stronger statement is required:

$$\mathbf{r}(\Phi)(\mathcal{P} \circ g) \subseteq \mathbf{r}(\Phi)(\mathcal{P}) \circ \mathbf{map}_{X,\Phi(X)} g$$

Program Extraction Theorem

A program term is called a *data term* if it is built from () by pairing and injections.

A formula is is called a *data formula* if it contains no free predicate variables and every subformula which is the premise of an implication or of the form $\nu\Phi(\vec{t})$ is non-computational.

Theorem
From a proof of a data formula A, one can extract a program term M with the property that M reduces to a data term provably realising A.

Remarks on related formal approaches

- Tatsuta defines **q**-realisability for coinduction and treats quantifiers differently. He uses realisability to extract witnesses for existential quantifiers while we we directly extract witnesses for inductive and coinductive predicates.
- Escardó's Real PCF is a typed λ-calculus while our (and Tatsuta's) realisers are untyped. Real PCF is intended to be used as a programming language by humans. Real PCF terms can be viewed as fragments of proofs. Can Real PCF be enriched to a full proof calculus?

The realisers of C_{AV}

In order to get an intuitive understanding of realisability one can assign *types* to realisers.

Recall:

$$\begin{aligned} \mathrm{AV} &= \{\mathrm{av}_d \mid d \in \mathrm{SD}\} \\ \mathrm{C}_{\mathrm{AV}} &= \nu X.\{d(x) \mid d \in \mathrm{AV}, x \in X\} \end{aligned}$$

The type of realisers of $\mathrm{C}_{\mathrm{AV}}(x)$ is

$$\mathrm{SD}^{\omega} := \nu\alpha.\mathrm{SD} \times \alpha$$

The realisers of $\mathrm{C}_{\mathrm{AV},\mathrm{AV}}$

Recall:

$$\begin{aligned}\mathrm{C}_{\mathrm{AV},\mathrm{AV}} \quad = \quad & \nu F \,.\, \mu G \,. \\ & \{e \circ f : \mathbb{I} \to \mathbb{I} \mid e \in \mathrm{AV}, f \in F\} \cup \\ & \{h : \mathbb{I} \to \mathbb{I} \mid \forall d \in \mathrm{AV}\ h \circ \mathrm{av}_d \in G\}\end{aligned}$$

The type of realisers of $\mathrm{C}_{\mathrm{AV}}(f)$ is

$$\nu\alpha \,.\, \mu\beta \,.\, \mathrm{SD} \times \alpha + \beta^3$$

$\Rightarrow$ Hancock, Pattinson, Ghani.

Understanding $\nu\alpha\,.\,\mu\beta\,.\,\mathrm{SD}\times\alpha+\beta^3$

Define T as the largest solution of the domain equation

$$T = \mathrm{SD} \times T + T^3$$

i.e. the elements of T are non-wellfounded trees with two kinds of nodes:

- **Writing nodes:** $W(d, t)$ where $d \in \mathrm{SD}$ and $t \in T$.
- **Reading nodes:** $R(t_{-1}, t_0, t_1)$ where $t_i \in T$.

$\nu\alpha\,.\,\mu\beta\,.\,\mathrm{SD}\times\alpha+\beta^3$ corresponds to the set of those trees in T that have on every infinite path infinitely many writing nodes.

Metric spaces

A *metric space* $X = (X, \sigma, P)$ consists of

- a set X,
- a metric σ on X,
- a dense set $P \subseteq X$ (of *concrete* elements).

For a rational number $\epsilon > 0$ and $p \in P$ we define

$$\mathrm{B}_\epsilon(p) := \{x \in X \mid \sigma(p, x) \leq \epsilon\}$$

For convenience, we only work with metric spaces that are *bounded*, i.e. $X \subseteq \mathrm{B}_M(p)$ for some $M > 0$ and $p \in P$.

Uniform continuity

Let $X = (X, P, \sigma)$ and $Y = (Y, Q, \tau)$ be metric spaces.

Let δ, ϵ range over positive rational numbers and x, x' over X.

Recall that a function $f : X \to Y$ is **uniformly continuous (u.c.)** if

$$\forall \epsilon \, \exists \delta \, \forall x, x' \in X$$

$$\sigma(x, x') \leq \delta \;\Rightarrow\; \tau(f(x), f(x')) \leq \epsilon$$

The computational content of this definition is a "backwards function" that computes the δ from the ϵ.

This backwards function is neither sufficient nor useful for efficient implementations of u.c. functions.

Therefore, we work with a different definition of uniform continuity.

m-continuity

Let again $X = (X, P, \sigma)$ and $Y = (Y, Q, \tau)$ be metric spaces and let p range over P, a, b range over $\mathbb{R}^+$, and set $\hat{a} := \{\delta \in \mathbb{Q}^+ \mid \delta \leq a\} = (0, a] \cap \mathbb{Q}$.

A **modulus** is a relation $m \subseteq \mathbb{Q}^+ \times \mathbb{Q}^+$ such that

$$\forall b\, \exists a \;\; m[\hat{a}] \subseteq \hat{b}$$

A relation $f \subseteq X \times Y$ is *m*-**continuous**, if

$$\forall \delta, p\, \exists \epsilon \in m(\delta)\, \exists q \;\; f[\mathrm{B}_\delta(p)] \subseteq \mathrm{B}_\epsilon(q)$$

Note that the realisers of *m*-continuity have type $\mathbb{Q} \times \mathbb{Q} \to \mathbb{Q} \times \mathbb{Q}$ and compute a “forward” information.

Lemma

A relation $f \subseteq X \times Y$ is *m*-continuous for some modulus m iff it is a partial function which is uniformly continuous on its domain.

Lipschitz conditions and contractivity

For $\lambda \geq 0$ we define

$$m_\lambda := \{(\delta, \epsilon) \mid \epsilon \leq \lambda\delta\}$$

Clearly, m_λ is a modulus and $m_\lambda \circ m_\gamma = m_{\lambda\gamma}$.

A relation $f \subseteq X \times Y$ is called *Lipschitz* if it is m_λ-continuous for some $\lambda \geq 0$. If $\lambda < 1$, we call f *contracting*.

Lemma A relation $f \subseteq X \times Y$ is λ-continuous iff it is a partial function and $\tau(f(x), f(x')) \leq \lambda \cdot \sigma(x, x')$ for all $x, x' \in \mathrm{dom}(f)$. Hence a function is Lipschitz iff it is Lipschitz in the usual sense.

Metric digit spaces

A *metric digit space* $X = (X, \sigma, P, D)$ is a metric space (X, σ, P) together with a set of digits $D \subseteq X \to X$.

(X, σ, P, D) is called

- *contracting* if there is $\lambda < 1$ such that all $d \in D$ are contracting with factor λ.
- *invertible* if d^{-1} is u.c. for all $d \in D$.
- *covering* if there is an $\epsilon_0 > 0$ such that, for all $\epsilon > 0$ and $p \in P$, either there exists $d \in D$ with $\mathrm{B}_\epsilon(p) \subseteq d[X]$, or $\epsilon > \epsilon_0$.
- *finitely covering* if there is a finite subset of D which is uniformly covering.

Example: $(\mathbb{I}, \mathrm{AV})$ has all these properties.

Characterisation of uniform continuity

Characterisation Lemma

Let $X = (X, \sigma, P, D)$ and $Y = (Y, \tau, Q, E)$ be metric digit spaces.
Set $\mathrm{U} := \{f : X \to Y \mid f \text{ u.c.}\}$ and $\mathrm{C} := \mathrm{C}_{D,E}$.

(a) If X is contracting, and Y is invertible and covering, then $\mathrm{U} \subseteq \mathrm{C}$.

(b) Assume D is finite. If X is invertible and finitely covering, and Y is contracting, then $\mathrm{C} \subseteq \mathrm{U}$.

Corollary (change of digits)
Let (X, σ, P) be a metric space and let $D, E \subseteq X \to X$. If D is contracting, and E is invertible and covering, then $\mathrm{C}_D \subseteq \mathrm{C}_E$.

Proof
The identity function on X is u.c. and hence in $\mathrm{C}_{D,E}$, by (a).

Iterated maps

The family of logistic maps (transformed from $[0, 1]$ to $\mathbb{I} = [-1, 1]$):

$$f_a : \mathbb{I} \to \mathbb{I}, \quad f(x) = a * (1 - x^2) - 1 \qquad (0 \leq a \leq 2).$$

f_a is $2a$-Lipschitz, hence uniformly continuous (Lipschitz Lemma), hence in $\mathrm{C} := \mathrm{C}_{\mathrm{AV},\mathrm{AV}} \subseteq \mathbb{I} \to \mathbb{I}$ (Characterisation Lemma (a)).

Hence the iterated maps f_a^n are in C (Composition Lemma) and therefore define signed digit stream transformers (Application Lemma).

The extracted program computes 100 digits of $f_2^n(2/3)$, where $n \leq 600$, in approximately 15 minutes (GHC).

The main point of this example is to demonstrate the **memoizing effect** of the tree representation of u.c. functions (see also Hinze, Altenkirch).

Similar case studies were carried out by Escardó, Plume, Marcial–Romero, and Blanck (up to 10^6 iterations).

π

For the metric digit space $(\mathbb{I}, \mathrm{AV})$ we have $\pi/4 \in \mathrm{C}_D$.

Proof We use the formula

$$\frac{\pi}{4} = \frac{1}{2} + \frac{1}{3}\left(\frac{1}{2} + \frac{2}{5}\left(\frac{1}{2} + \frac{3}{7}\left(\frac{1}{2} + \frac{4}{9}\left(\frac{1}{2} + \dots\right)\right)\right)\right)$$

i.e. $\pi/4 = f_0(f_1(\dots))$ where

$$f_n(x) := \frac{1}{2} + \frac{n\,x}{2n+1}.$$

Hence we have $\pi/4 \in \mathrm{C}_F$ where $F := \{f_n \mid n \in \mathbb{N}\}$. Since F is contracting and AV is invertible and covering, it follows, by the Change of Digits L., $\pi/4 \in \mathrm{C}_D$.

Integration

For a continuous function $f : \mathbb{I} \to \mathbb{R}$ we set
$\int f := \int_{-1}^{1} f = \int_{-1}^{1} f(t)\,\mathrm{d}t \in \mathbb{R}$.

Lemma

(a) $\int(\mathrm{av}_i \circ f) = \mathrm{av}_{2 \cdot i}(\int f)$
(b) $\int f = \frac{1}{2}(\int(f \circ \mathrm{av}_{-1}) + \int(f \circ \mathrm{av}_1))$.

(see also Escardó, Simpson, Scriven)

Integration Lemma

Let (X, σ, P, D) be a covering and invertible metric digit system and $f \in \mathrm{C}_{D \otimes \mathrm{AV}, \mathrm{AV}}$. Then the function mapping $(a, b, x) \in \mathbb{I}^2 \times X$ to $\int_a^b f(x,t)\,\mathrm{d}t$ is well-defined and uniformly continuous.

Conclusion

- "Proofs as programs" deserves a "last chance".
- New (correct!) programs extracted that would have been difficult to "guess".
- Using a fine tuning of realisability it is possible to do abstract mathematics as usual, and still get computational content.
- A viable approach to GC6 (Dependable systems evolution, verifying compiler)?

Further work

- Clarify connections with related work.
- Implement and automate (joint work with the Munich logic group (Minlog) and Anton Setzer (Agda)).
- Overcome limitations: no finite system of contracting and uniformly covering digits exists on the compact metric space of non-empty compact sets with the Hausdorff metric (joint work with Dieter Spreen).
- Power series via higher type digits.
- "Realiser sensitive" logic:
  $\forall x\,(A(x) \to B(x) \to C(x))$ vs.
  $\forall x\,(A(x) \to \neg B(x) \vee C(x))$ for decidable $B(x)$.

References

T. Altenkirch.
Representations of first order function types as terminal coalgebras. TLCA 2001. LNCS 2044, 8–21, 2001.

A. Abel.
Syntactical normalization for intersection types with term rewriting rules. HOR'07, 2007.

B.
Strong normalization for applied lambda calculi. *LMCS*, 1(2):1–14, 2005.

T. Hou, B.
Coinduction for Exact Real Number Computation. *ToCS*, 2007.

References

Y. Bertot.
Coinduction in Coq. In Lecture Notes of TYPES Summer School 2005, August 15-26 2005, Sweden, vol. II (2005).

J. Blanck.
Efficient exact computation of iterated maps. *JLAP*, 64:41–59, 2005.

J. Blanck.
Exact real arithmetic using centred intervals and bounded error terms. *JLAP*, 66:50–67, 2006.

W. Buchholz.
A term calculus for (co-)recursive definitions on streamlike data-structures. *APAL*, 136:75–90, 2005.

References

A. Ciaffaglione, P. Di Gianantonio, Di P.
A certified, corecursive implementation of exact real numbers.
TCS 351:39–51, 2006.

T. Coquand, A. Spiwack.
Proof of strong normalisation using domain theory. *Lics 2006*.

A. Edalat, R. Heckmann.
Computing with real numbers - I. The LFT approach to real number computation - II. A domain framework for computational geometry. International summer school on applied semantics, Caminha, Portugal, Springer, 193–267, 2002.

A. Edalat, P.J. Potts, P. Sünderhauf.
Lazy computation with exact real numbers. Proceedings of the Third ACM SIGPLAN International Conference on Functional Programming, 185-194, 1998.

References

M.H. Escardó.
PCF extended with real numbers. *TCS* 162:79–115, 1996.

M.H. Escardó, A. Simpson.
A universal characterization of the closed Euclidean interval. *LICS 2001*, 115-125.

J. Raymundo Marcial–Romero, M.H. Escardó.
Semantics of a sequential language for exact real-number computation. *TCS* 379:120–141, 2007.

P. Hancock, D. Pattinson, N. Ghani.
Representation of stream processors using nested fixed points. *unpublished*. 2008.

References

R. Hinze.
Memo functions, polytypically!". Proceedings of the Second Workshop on Generic Programming, WGP 2000, Ponte de Lima, Portugal. 2000.

B. Jacobs, J. Rutten.
A Tutorial on (Co)Algebras and (Co)Induction. EATCS Bulletin 62, 222–259, 1997.

R. O'Connor.
Certified Exact Transcendental Real Number Computation in Coq. Unpublished. 2008

R. O'Connor, B. Spitters.
A computer verified monadic, functional implementation of the integral. Unpublished. 2008

References

M. Niqui.
Formalising exact arithmetic in type theory. CiE 2005: New Computational Paradigms. Amsterdam, LNCS **3526** (2005) 368–377.

M.H. Escardó, D. Pavlovic.
Calculus in coinductive form. School of Cognitive and Computing Sciences, University of Sussex. Technical Report 97:05, 1997.

D. Pavlovic, V. Pratt.
The continuum as a final coalgebra. *TCS* 280:105–122, 2002.

D. Plume.
A Calculator for Exact Real Number Computation. 4th year project. Departments of Computer Science and Artificial Intelligence, University of Edinburgh (1998).

References

G. D. Plotkin.
LCF considered as a programming language. *TCS*, 5:223–255, 1977.

P.J. Potts, A. Edalat and M.H. Escardó.
Semantics of exact real number arithmetic. *Lics 1997*.

M. Tatsuta.
Realizability of Monotone Coinductive Definitions and Its Application to Program Synthesis. *Proceedings of Fourth International Conference on Mathematics of Program Construction*, LNCS 1422 (1998) 338–364.