Consistent ultrafinitist logic

LCC 2022 presentation

Michał J. Gajda https://www.migamake.com

2022-02-19

Plan

Plan

- Background

Plan

- Background
  - Transfinitist mathematics

Plan

- Background
  - Transfinitist mathematics
  - Finitism

Plan

- Background
  - Transfinitist mathematics
  - Finitism
  - Ultrafinitism

Plan

- Background
  - Transfinitist mathematics
  - Finitism
  - Ultrafinitism
  - Ultraconstructivism

Plan

- Background
  - Transfinitist mathematics
  - Finitism
  - Ultrafinitism
  - Ultraconstructivism
- Logic

Plan

- Background
  - Transfinitist mathematics
  - Finitism
  - Ultrafinitism
  - Ultraconstructivism
- Logic
  - Ultrafinitist Logic syntax

Plan

- Background
  - Transfinitist mathematics
  - Finitism
  - Ultrafinitism
  - Ultraconstructivism
- Logic
  - Ultrafinitist Logic syntax
  - Simplifying positive polynomials

Plan

- Background
  - Transfinitist mathematics
  - Finitism
  - Ultrafinitism
  - Ultraconstructivism
- Logic
  - Ultrafinitist Logic syntax
  - Simplifying positive polynomials
  - CLF inference rules

Plan

- Background
  - Transfinitist mathematics
  - Finitism
  - Ultrafinitism
  - Ultraconstructivism
- Logic
  - Ultrafinitist Logic syntax
  - Simplifying positive polynomials
  - CLF inference rules
- Consequences

Plan

- Background
  - Transfinitist mathematics
  - Finitism
  - Ultrafinitism
  - Ultraconstructivism
- Logic
  - Ultrafinitist Logic syntax
  - Simplifying positive polynomials
  - CLF inference rules
- Consequences
  - redefining decidability

Plan

- Background
  - Transfinitist mathematics
  - Finitism
  - Ultrafinitism
  - Ultraconstructivism
- Logic
  - Ultrafinitist Logic syntax
  - Simplifying positive polynomials
  - CLF inference rules
- Consequences
  - redefining decidability
  - axiomatic cryptography

Plan

- Background
  - Transfinitist mathematics
  - Finitism
  - Ultrafinitism
  - Ultraconstructivism
- Logic
  - Ultrafinitist Logic syntax
  - Simplifying positive polynomials
  - CLF inference rules
- Consequences
  - redefining decidability
  - axiomatic cryptography
  - philosophy of inference

Background

Transfinitist mathematics

Assumption of infinity:

- Numbers
- Enumerations

Finitism

- Finite proofs

Finitism

- Finite proofs
- Finite descriptions of proofs

Finitism

- Finite proofs
- Finite descriptions of proofs
- Examination of infinite proofs is unfeasible

Physical limits

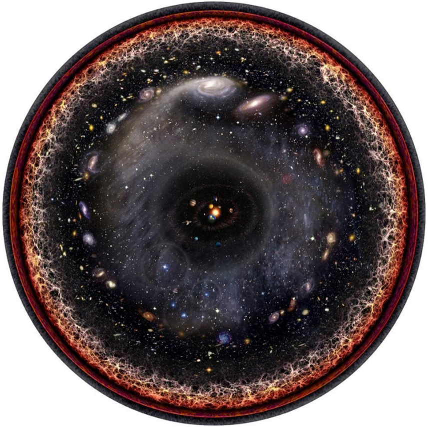

Figure 1: Observable universe in log scale

Ultrafinitism

- Feasible numbers (Sazonov 1995)

Ultrafinitism

- Feasible numbers (Sazonov 1995)
- Limit of observable universe (Lloyd 2002)

Ultrafinitism

- Feasible numbers (Sazonov 1995)
- Limit of observable universe (Lloyd 2002)
- Physical limits of accuracy (Gisin 2019)

Ultrafinitism

- Feasible numbers (Sazonov 1995)
- Limit of observable universe (Lloyd 2002)
- Physical limits of accuracy (Gisin 2019)
- Limit of computational density (Krauss and Starkman 2004)

Ultrafinitism

- Feasible numbers (Sazonov 1995)
- Limit of observable universe (Lloyd 2002)
- Physical limits of accuracy (Gisin 2019)
- Limit of computational density (Krauss and Starkman 2004)
- Gorelik-Bremermann limit (Gorelik 2010)

Bumpy road to ultrafinitism

- “no satisfactory developments exist” (Troelstra 1988)

Bumpy road to ultrafinitism

- “no satisfactory developments exist” (Troelstra 1988)
- Bounded Arithmetic (Krajicek 1995)

Bumpy road to ultrafinitism

- “no satisfactory developments exist” (Troelstra 1988)
- Bounded Arithmetic (Krajicek 1995)
- Primitive Recursive Functions are not all finitist functions (Schirn and Niebergall 2005)

Bumpy road to ultrafinitism

- “no satisfactory developments exist” (Troelstra 1988)
- Bounded Arithmetic (Krajicek 1995)
- Primitive Recursive Functions are not all finitist functions (Schirn and Niebergall 2005)
- naive finitism logic as inconsistent (Dummett 1975; Magidor 2007)

Proposed: ultraconstructivism

- consider only constructive developments

Proposed: ultraconstructivism

- consider only constructive developments
- carry explicit computational bounds

Proposed: ultraconstructivism

- consider only constructive developments
- carry explicit computational bounds
- check that proof exists within those bounds

Consistent Ultrafinitist Logic

Syntax

Sizes

Size variables:	v	$\in$	V
Positive naturals:	i	$\in$	$\mathbb{N} \setminus \{0\}$

Cost polynomials

$$n \geq 1$$

$$\begin{array}{lcll} \text{Polynomials:} & \rho & ::= & v|i|\rho+\rho|\rho*\rho|\rho^{\rho}|\,\mathit{iter}(\rho,\rho,v)|\rho[\![{}^{v}/_{\rho}]\!] \\ \text{Data size bounds:} & \alpha & ::= & \rho \\ \text{Computation bounds:} & \beta & ::= & \rho \end{array}$$

Iterated function composition: $\mathit{iter}(\rho_1, \rho_2, v)$ Function ρ_1 with a bound variable v is iterated ρ_2 times.

Types

Type or term variables:	x $\in$	X
Types:	τ $::=$	$v \big\vert \tau \wedge \tau \big\vert \tau \vee \tau \big\vert \forall x_v : \tau \rightarrow^{\alpha}_{\beta} \tau \big\vert \bot \big\vert \circ$
Environments:	Γ $::=$	$v_1 : {\tau^1}_{\beta_1}, ..., {\tau^n}_{\beta_n}$
Judgements:	J $::=$	$\Gamma \vdash^{\alpha}_{\beta} E : \tau$

x_v - term variable x with its size bound by size variable v

Notation $\forall x_v : A \implies^{\alpha(v)}_{\beta(v)} B$ binds proof variable x with type of A, and then bound in polynomials $\alpha(v)$ for complexity and $\beta(v)$ for depth of the normalized term.

Proof terms

$$\text{Terms:} \quad E \quad ::= \quad v \big| \lambda v.E \big| in_r(E) \big| in_l(E) \big| (E, E) \big| \, () \\ \quad \Big| \quad case\; E\; of \; \begin{array}{lcl} in_l(v) & \rightarrow & E; \\ in_r(v) & \rightarrow & E; \end{array}$$

β_i is an upper bound on depth of term proving A^i.

Judgements

$$\begin{array}{llll}
\text{Type or term variables:} & x & \in & X \\
\text{Types:} & \tau & ::= & v \big| \tau \wedge \tau \big| \tau \vee \tau \big| \forall x_v : \tau \to^{\alpha}_{\beta} \tau \big| \bot \big| \circ \\
\text{Environments:} & \Gamma & ::= & v_1 : {\tau^1}_{\beta_1}, ..., {\tau^n}_{\beta_n} \\
\text{Judgements:} & J & ::= & \Gamma \vdash^{\alpha}_{\beta} E : \tau
\end{array}$$

:w x_v - variable its size bound variable α_i is an upper bound on computation steps needed to evaluate A^i. β_i is an upper bound on depth of term proving A^i.

Rules

Variables

$$\frac{\Gamma \vdash^{\alpha}_{\beta} y_{\beta} : A \quad v \in V}{\Gamma, x_{\beta} : A \vdash^{1}_{\beta} x : A} \; \textit{var}$$

Unit type

$$\frac{}{\Gamma \vdash^{1}_{\beta} () : \circ} \; \textit{unit}$$

Subsumption

$$\frac{\Gamma \vdash^{\alpha_1}_{\beta_1} e : A \qquad \alpha_1 \leq \alpha_2 \qquad \beta_1 \leq \beta_2}{\Gamma \vdash^{\alpha_2}_{\beta_2} e : A} \; \textit{subsume}$$

Positive polynomials for easy bounds

Positive polynomials for easy bounds

$$x, y, e, f, g, h \geq 1$$

$$a, b \geq 0$$

$$\begin{array}{lrcl}
(1) & a * x^e + b * x^f & \leq & (a+b) * x^f \\
(2) & a * x^e * y^g & \leq & a * x^f * y^h \\
(3) & iter(e, g, x) & \leq & iter(f, h, x) \\
(4) & iter(a * x, e, x) & = & a^e * x \\
(5) & iter(x + a, e, x) & = & x + a * e \\
(6) & iter(x^e, g, x) & = & x^{e^g}
\end{array}$$

Conjunction

$$\frac{\Gamma \vdash^{\alpha_1}_{\beta_1} a^1 : A^1 \quad \Gamma \vdash^{\alpha_2}_{\beta_2} a^2 : A^2}{\Gamma \vdash^{\alpha_1+\alpha_2}_{\max(\beta_1,\beta_2)+1} (a^1, a^2) : A^1 \wedge A^2} \; \mathit{pair}$$

$$\frac{\Gamma \vdash^{\alpha}_{\beta+1} e : A^1 \wedge A^2 \quad i \in \{1,2\}}{\Gamma \vdash^{\alpha+1}_{\beta} \mathit{prj}_i \; e : A^i} \; \mathit{prj}_i$$

Alternative

$$\frac{\Gamma \vdash_{\beta}^{\alpha} e : A^{i} \quad i \in \{l, r\}}{\Gamma \vdash_{\beta+1}^{\alpha+1} in_{i}(e) : A^{1} \vee A^{2}} \; inj$$

$$\frac{\Gamma \vdash_{\beta_{\vee}+1}^{\alpha_{\vee}} a : A^{1} \vee A^{2} \qquad \Gamma, x : {A^{1}}_{\beta_{\vee}} \vdash_{\beta_{1}}^{\alpha_{1}} b : B \qquad \Gamma, y : {A^{2}}_{\beta_{\vee}} \vdash_{\beta_{2}}^{\alpha_{2}} c : B}{\Gamma \vdash_{\max(\beta_{1},\beta_{2})}^{\alpha_{\vee}+max(\alpha_{1},\alpha_{2})+1} case\ a\ of\ \begin{array}{lll} in_{l}(x) & \rightarrow & b; \\ in_{r}(y) & \rightarrow & c; \end{array} : B} \; case$$

Abstraction and application

$$\frac{\Gamma, x_v : A \vdash^{\alpha(v)}_{\beta(v)} e : B}{\Gamma \vdash^{\alpha(1)+1}_{\beta(1)+1} \lambda x.e : \forall a_v : A \rightarrow^{\alpha(v)}_{\beta(v)} B} \; abs$$

$$\frac{\Gamma \vdash^{\alpha_1}_{\beta_1} e : \forall a : A_v \rightarrow^{\alpha_2(v)}_{\beta_2(v)} B \quad \Gamma \vdash^{\alpha_3}_{\beta_3} a : A}{\Gamma \vdash^{\alpha_1+\alpha_2(\beta_3)+\alpha_3}_{\beta_2(\beta_3)} e\ a : B} \; app$$

Please note that notation $\forall x_v : A \rightarrow^{\alpha(v)}_{\beta(v)} B$ has a size variable v declared as a depth of term variable x, and then bound in polynomials $\alpha(v)$ and $\beta(v)$

The notation $\alpha(1)$ is a shortcut for $\alpha[\![1/v]\!]$ in the rules *abs* and *app*.

Recursion

$$\frac{\Gamma \vdash^{\alpha_1}_{\beta_1} f : A_v \rightarrow^{\alpha_2(v_1)}_{\beta_2(v_2)} A \quad \Gamma \vdash^{\alpha_3}_{\beta_3} k : B \quad \Gamma \vdash^{\alpha_4}_{\beta_4} a : A}{\Gamma \vdash^{\alpha_1 + \alpha_3 + iter(\alpha_2, \beta_3, v_1)[\![\beta_4 / v_1]\!] + \alpha_4 + 1}_{\beta_1 [\![iter(\beta_2, \beta_3, v_2)[\![\beta_4/v_2]\!] / v]\!]} rec(f, k, a) : B} \; rec$$

$rec(f, k, a)$ iterates function f at k times over a.

Consistency

Consistency

- Every rule beside *subsume* and *rec* is present in intuitionistic logic.

Consistency

- Every rule beside *subsume* and *rec* is present in intuitionistic logic.
- *rec fka* can be understood as k unfoldings of *app*: $f(f(..(a)))$

Consistency

- Every rule beside *subsume* and *rec* is present in intuitionistic logic.
- *rec fka* can be understood as k unfoldings of *app*: $f(f(..(a)))$
- Hence consistency by embedding in IL: all *instance* of the statements can be reduced to finite IL proofs

Expressivity

Expressivity

- Can express bounded loop programs (Meyer and Ritchie 1967)

Expressivity

- Can express bounded loop programs (Meyer and Ritchie 1967)
- Loop programs are primitive-recursive-complete

Expressivity

- Can express bounded loop programs (Meyer and Ritchie 1967)
- Loop programs are primitive-recursive-complete
- Bounds computable in advance - at least primitive recursive

Expressivity

- Can express bounded loop programs (Meyer and Ritchie 1967)
- Loop programs are primitive-recursive-complete
- Bounds computable in advance - at least primitive recursive
- Turing machine up to bounds

Emulation completeness

Emulation completeness

Theorem

Assume a time complexity $c(x)$ for program (or proof) s that can be encoded as CUFL bounds. Iff we can emulate (encode evaluation) of $f(x)$ with an overhead e for each step, then we can prove that complexity of evaluating s is $e * c(x) + cc(x)$.

Turing machine step can be easily emulated in $log^2(|\Sigma|)$.

Proof of emulation completeness

Proof of emulation completeness

Proof.

Assuming that $e(f)$ is function emulation in CUFL, we can write proof expression $iter(e(f), e(c), x)$. This expression evaluated encoded s and has exactly the assumed complexity

$\square$

Meta-reasoning

Meta-reasoning

We can encode bounds as terms:

$$
\begin{array}{lcl}
\mathsf{Var}_\beta & = & \mathsf{Nat}_\beta \\
\mathsf{Bound}_{\beta+1} & = & \mathsf{Var} \vee \mathsf{Nat}_\beta \vee \circ \vee (\mathsf{Bound}_\beta, \mathsf{Bound}_\beta) \\
 & \vee & (\mathsf{Bound}_\beta, \mathsf{Bound}_\beta) \\
 & \vee & (\mathsf{Bound}_\beta, \mathsf{Bound}_\beta) \vee (\mathsf{Bound}_\beta, (\mathsf{Bound}_\beta, \mathsf{Var})) \\
 & \vee & (\mathsf{Bound}_\beta, (\mathsf{Var}, \mathsf{Bound}_\beta)) \\
[\![v]\!] & = & in_l(in_l(in_l(\mathbb{B}\,(v)))) \\
[\![i]\!] & = & in_l(in_l(in_r(i))) \\
[\![()]\!] & = & in_l(in_l(in_r(()))) \\
[\![\rho_1 + \rho_2]\!] & = & in_l(in_r(in_r(([\![\rho_1]\!], [\![\rho_2]\!])))) \\
[\![\rho_1 * \rho_2]\!] & = & in_r(in_l(in_l(([\![\rho_1]\!], [\![\rho_2]\!])))) \\
[\![\rho_1^{\rho_2}]\!] & = & in_r(in_l(in_r(([\![\rho_1]\!], [\![\rho_2]\!])))) \\
[\![iter(\rho_1, \rho_2, v)]\!] & = & in_r(in_r(in_l(([\![\rho_1]\!], ([\![\rho_2]\!], \mathbb{B}\,(v)))))) \\
[\![\rho_1\ [\![\rho/v]\!]]\!] & = & in_r(in_r(in_r(([\![\rho_1]\!], ([\![\rho_2]\!], \mathbb{B}\,(v))))))
\end{array}
$$

Meta-reasoning 2

Meta-reasoning 2

We can also encode types:

$$
\begin{array}{lcl}
[\![A \vee B]\!] & = & in_l(in_l(([\![A]\!], [\![B]\!]))) \\
[\![A \wedge B]\!] & = & in_l(in_r(([\![A]\!], [\![B]\!]))) \\
[\![\forall x_v : A \rightarrow^{\alpha}_{\beta} B]\!] & = & in_r(in_l((\lambda x : A.[\![B]\!], (\lambda v : \mathsf{Nat}_v .[\![\alpha]\!], \lambda v : \mathsf{Nat}_v .[\![\beta]\!])))) \\
[\![\circ]\!] & = & in_r(in_r(()))
\end{array}
$$

Meta-reasoning 3

Meta-reasoning 3

Finally we can encode the proof terms:

$$
\begin{array}{lcl}
[\![x_v]\!] & = & in_l(in_l(in_l(in_l((\mathbb{B}(x), v))))) \\
[\![subsume(A, B)]\!] & = & in_l(in_l(in_l(in_r(([\![B]\!]_{\mathsf{Bound}}, [\![A]\!]))))) \\
[\![unit]\!] & = & in_l(in_l(in_r(in_l(())))) \\
[\![in_l(A)]\!] & = & in_l(in_l(in_r(in_r(A)))) \\
[\![in_r(A)]\!] & = & in_l(in_r(in_l(in_l(A)))) \\
[\![prj_l A]\!] & = & in_l(in_r(in_l(in_r(A)))) \\
[\![prj_r A]\!] & = & in_l(in_r(in_r(in_l(A)))) \\
[\![(A, B)]\!] & = & in_l(in_r(in_r(in_r(([\![A]\!], (interpB, ,)))))) \\
[\![app(A, B)]\!] & = & in_r(in_l(in_l(in_l(([\![A]\!], [\![B]\!]))))) \\
[\![abs\ \lambda x_v.A]\!] & = & in_r(in_l(in_l(in_r(((\mathbb{B}(x), \mathbb{B}(v)), [\![A]\!]))))) \\
[\![rec]\!] & = & in_r(in_l(in_r(in_l((([\![A]\!], [\![B]\!]), \mathbb{B}(v))))))
\end{array}
$$

Meta-reasoning 4

Meta-reasoning 4

Theorem

Emulation completeness of ULF can be proven in itself.

Problems stated

Problems stated

1. First we need to compute bound is within our limit

Problems stated

1. First we need to compute bound is within our limit
2. Then we are guaranteed that we have an answer within given time.

Problems stated

1. First we need to compute bound is within our limit
2. Then we are guaranteed that we have an answer within given time.
3. We may likewise bound computation of bounds.

Avoiding infinities

Avoiding infinities

- use convergent series instead of reals (Zeilberger 2010)

Avoiding infinities

- use convergent series instead of reals (Zeilberger 2010)
- statements about finite descriptions instead of infinite series

Avoiding infinities

- use convergent series instead of reals (Zeilberger 2010)
- statements about finite descriptions instead of infinite series
- avoids *implicit* infinities in statement of the problem

Avoiding infinities

- use convergent series instead of reals (Zeilberger 2010)
- statements about finite descriptions instead of infinite series
- avoids *implicit* infinities in statement of the problem
- for example: every statement about **all** natural numbers is statement about infinity

Avoiding infinities

- use convergent series instead of reals (Zeilberger 2010)
- statements about finite descriptions instead of infinite series
- avoids *implicit* infinities in statement of the problem
- for example: every statement about **all** natural numbers is statement about infinity
- avoid transfinite ordinals

Consequences

Consequences

- Bounded time-to-answer (*when*)

Consequences

- Bounded time-to-answer (*when*)
- Redefines logical expressivity

Consequences

- Bounded time-to-answer (*when*)
- Redefines logical expressivity
- Redefines decidability (what **and** when)

Consequences

- Bounded time-to-answer (*when*)
- Redefines logical expressivity
- Redefines decidability (what **and** when)
- Only computable functions

Consequences

- Bounded time-to-answer (*when*)
- Redefines logical expressivity
- Redefines decidability (what **and** when)
- Only computable functions
- Avoids semidecidability paradox

Future work

Future work

- Expressiveness

Future work

- Expressiveness
- List paradoxes reverted

Future work

- Expressiveness
- List paradoxes reverted
- Theorem prover

Conclusion

Conclusion

- Ultrafinitism is consistent

Conclusion

- Ultrafinitism is consistent
- Consistency by embedding in intuitionistic logic

Conclusion

- Ultrafinitism is consistent
- Consistency by embedding in intuitionistic logic
- Bounded time-to-answer

Conclusion

- Ultrafinitism is consistent
- Consistency by embedding in intuitionistic logic
- Bounded time-to-answer
- Semidecidability as paradox

References

References I

Dummett, Michael. 1975. "Wang's Paradox." *Synthese* 30 (3/4): 301–24. http://www.jstor.org/stable/20115034.

Gisin, Nicolas. 2019. "Indeterminism in Physics, Classical Chaos and Bohmian Mechanics. Are Real Numbers Really Real?" https://arxiv.org/abs/1803.06824.

Gorelik, Gennady. 2010. "Bremermann's Limit and cGh-Physics." https://arxiv.org/abs/0910.3424.

Krajicek, Jan. 1995. *Bounded Arithmetic, Propositional Logic and Complexity Theory*. Encyclopedia of Mathematics and Its Applications. Cambridge University Press. https://doi.org/10.1017/CBO9780511529948.

Krauss, Lawrence, and Glenn Starkman. 2004. "Universal Limits on Computation," May.

Lloyd, Seth. 2002. "Computational Capacity of the Universe." *Physical Review Letters* 88 23: 237901.

Magidor, Ofra. 2007. "Strict Finitism Refuted?" *Proceedings of the Aristotelian Society* 107 (1pt3): 403–11. https://doi.org/10.1111/j.1467-9264.2007.00230.x.

References II

Meyer, Albert R., and Dennis M. Ritchie. 1967. “The Complexity of Loop Programs.” In *Proceedings of the 1967 22nd National Conference*, 465–69. ACM ’67. New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/800196.806014.

Sazonov, Vladimir Yu. 1995. “On Feasible Numbers.” In *Logic and Computational Complexity*, edited by Daniel Leivant, 30–51. Berlin, Heidelberg: Springer Berlin Heidelberg.

Schirn, Matthias, and Karl-Georg Niebergall. 2005. “Finitism = PRA? On a Thesis of w. W. Tait.” *Reports on Mathematical Logic*, January.

Troelstra, A. S. 1988. *Constructivism in Mathematics: An Introduction*. Elsevier. https://www.sciencedirect.com/bookseries/studies-in-logic-and-the-foundations-of-mathematics/vol/121/suppl/C.

Zeilberger, Doron. 2010. “Opinion 108: ...the Feeling Is Mutual: I Feel Sorry for Infinitarian Hugh Woodin for Feeling Sorry for Finitists Like Myself! (And the "Lowly" Finite Is MUCH More Beautiful Than Any "Infinite").” 2010. https://sites.math.rutgers.edu/~zeilberg/Opinion108.html.