Inhabiting Kripke's truth via a working paracomplete formal arithmetic

Bryan Ford
EPFL – IC – DEDIS
bryan.ford@epfl.ch

Saul Kripke Center, CUNY – 23 March 2026

What to make of paradox?

- **The Liar:** “This statement is false”
- **The Truthteller:** “This statement is true”
- **Curry:** “If this statement is true then pigs fly”
- **Russell:**
  “the set of all sets not containing themselves”
- **Berry:**
  “the least number not nameable in ten words”

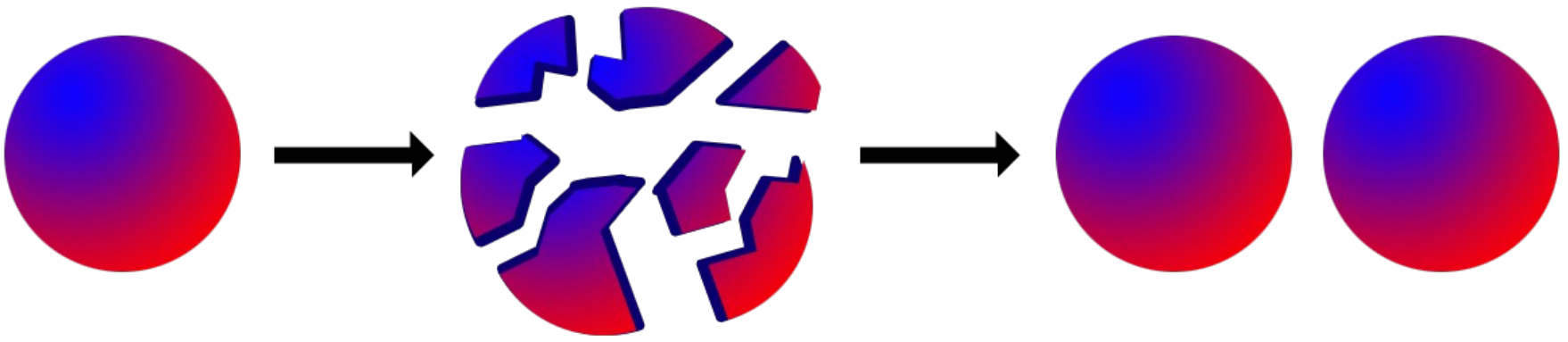
What to make of paradox?

Yablo:

1) All the following statements are false.
2) All the following statements are false.
3) All the following statements are false.
4) …

What to make of paradox?

Banach-Tarski: reassemble 1 unit ball into 2 balls



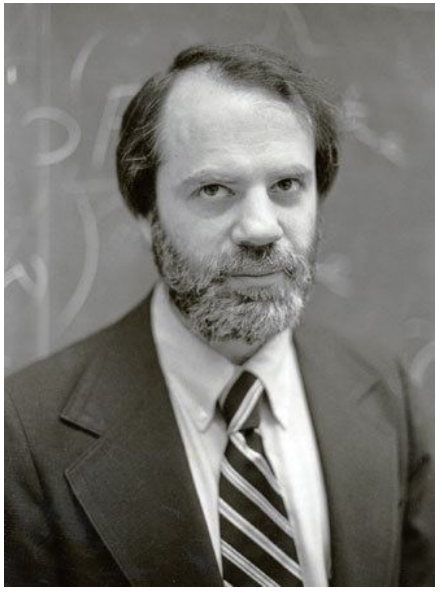
[image credit: cognitive coitus]

Talk Outline

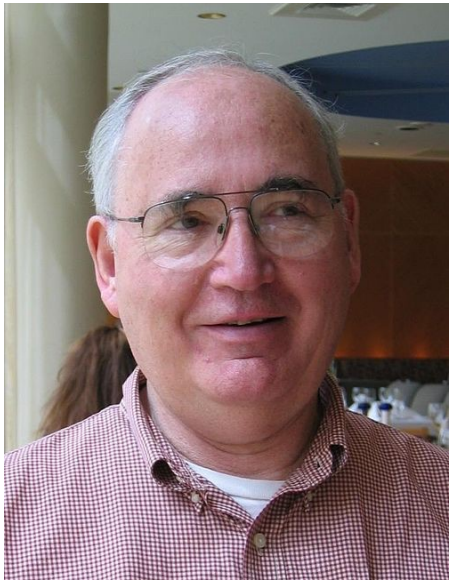
- **Background: whence this work?**
- Introduction: grounded arithmetic (GA)
- Habeas quid: enabling freedom of recursion
- Metatheory: what properties does GA have?
- Speculation: where might this lead us?

Gödel, Kripke, Scott

Kurt Gödel
(Logic)



Saul Kripke
(Philosophy)



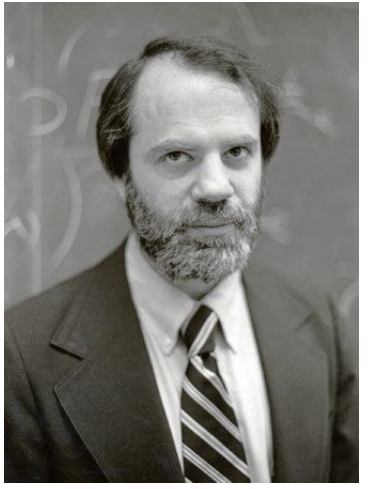
Dana Scott
(Computer Science)

Kripke on Truth (1975)

OUTLINE OF A THEORY OF TRUTH *

I. THE PROBLEM

EVER since Pilate asked, "What is truth?" (*John* XVIII, 38), the subsequent search for a correct answer has been inhibited by another problem, which, as is well known, also arises in a New Testament context. If, as the author of the Epistle to Titus supposes (*Titus* I, 12), a Cretan prophet, "even a prophet of their own," asserted that "the Cretans are always liars," and if "this testimony is true" of all other Cretan utterances, then it seems that the Cretan prophet's words are true if and only if they are false. And any treatment of the concept of truth must somehow circumvent this paradox.

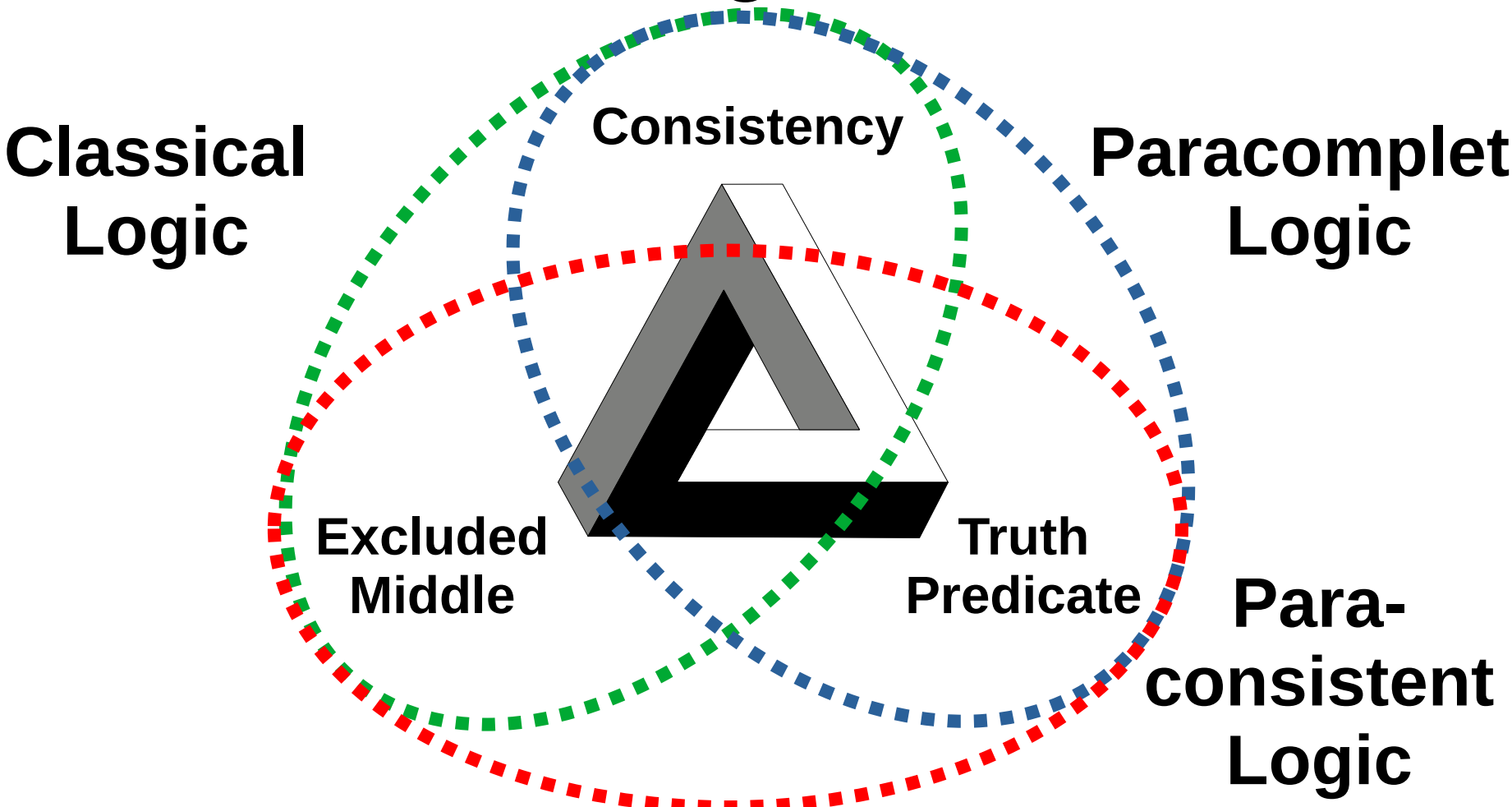


Kripke on Truth (1975)

Proposed a *paracomplete* approach to reasoning

- Assigns “truth” to statements iteratively, based on *grounds* for their being true or false
- Statements might fail to be either true or false: we must reject law of excluded middle (LEM)
- Liar, Truthteller, and many other paradoxes are just *ungrounded* statements of this kind.

A reasoning “trilemma”



Is Kripke-like reasoning *usable*?

Could we use it in practice for anything like *ordinary* reasoning about *ordinary* things?

- We need not just a logic but a *subject matter:* a useful domain of discourse to reason about
- Intuitionistic logic has Heyting arithmetic (HA), constructive analysis, modern proof assistants

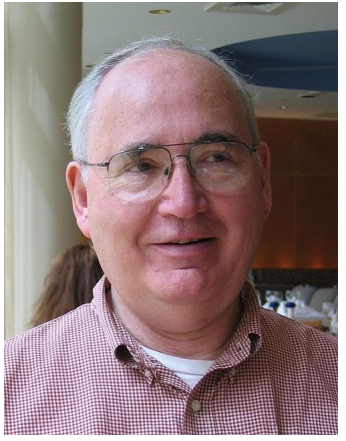
Perhaps a *grounded* theory of natural numbers?

Grounded truth as computation

Observation:

- Statements that Kripke assigns “true” or “false” are *computations* that *terminate* with “1” or “0”
- Statements that Kripke calls “ungrounded” are just *computations* that *never terminate*.
- Grounded truth is closely related to – perhaps *equivalent* to – the semantics of computation

Dana Scott's Domain Theory (1976)



DATA TYPES AS LATTICES

To the Memory of Christopher Strachey, 1916–1975

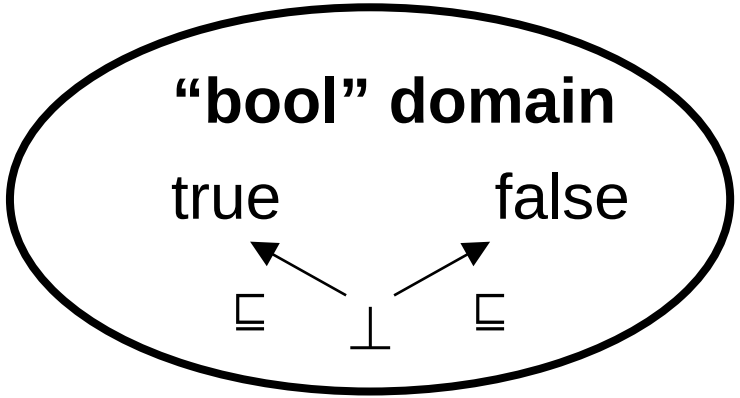
DANA SCOTT†

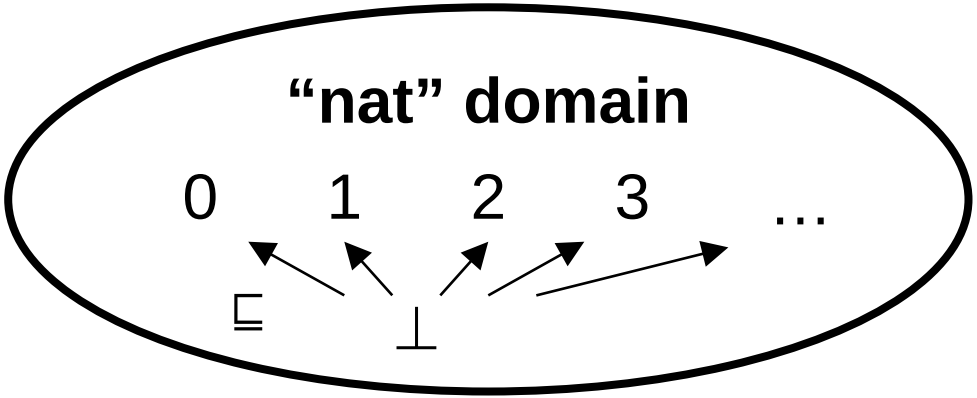
Abstract. The meaning of many kinds of expressions in programming languages can be taken as elements of certain spaces of "partial" objects. In this report these spaces are modeled in one universal domain $\mathbf{P}\omega$, the set of all subsets of the integers. This domain renders the connection of this semantic theory with the ordinary theory of number theoretic (especially general recursive) functions clear and straightforward.

Dana Scott's Domain Theory (1976)

Basis of *denotational semantics* of computation

- Assign set-theoretic "meaning" to data & code, *including* functions that need not terminate
- Derive *information approximations* via relation ⊑





Talk Outline

- Background: whence this work?
- **Introduction: grounded arithmetic (GA)**
- Habeas quid: enabling freedom of recursion
- Metatheory: what properties does GA have?
- Speculation: where might this lead us?

Broad kinda-readable overview

Reasoning Around Paradox with Grounded Deduction

Bryan Ford

arXiv preprint 2409.08243
September 12, 2024 (first version)

Abstract:

How can we reason around logical paradoxes without falling into them? This paper introduces grounded deduction or GD, a Kripke-inspired approach to first-order logic and arithmetic that is neither classical nor intuitionistic, but nevertheless appears both pragmatically usable and intuitively justifiable. GD permits the direct expression of unrestricted recursive definitions -- including paradoxical ones such as 'L := not L' -- while adding dynamic typing premises to certain inference rules so that such paradoxes do not lead to inconsistency. This paper constitutes a preliminary development and investigation of grounded deduction, to be extended with further elaboration and deeper analysis of its intriguing properties.

Grounded Arithmetic (GA): Goal

Define a *usable, working* paracomplete system

- Subject matter: natural numbers, computation
- Propositional ¬ ∧ ∨ → ↔ and quantifiers ∀ ∃
- Inference rules: infer *terminating* computations
- Safely *avoid* (“reason around”) nontermination: ungrounded statements, functions yielding ⊥

Grounded Arithmetic: non-goals

For now, GA is *not* immediately concerned with:

- Proving *nontermination*
- Real numbers, transfinite sets
- Non-computable mathematical objects
- What does strong “Not True” mean?

Let’s walk before trying to run!

What about (strong) “Not True”?

Weak “not true” ≡ “false”

- *L* ≡ “this statement is false” ≡ “…is not true” ≡ ⊥

Strong “Not True” would permit us to reason:

- “*L* is Not True and Not False” → “*L* is Not True”
- First step down “slippery slope” of *revenge* [Simmons, Maudlin, Beall, Field, …]
  - Not My Problem (for now)

The “freedom of recursion” problem

Casual programming

YES 

```
main() { printf(“Hi!”); }
main() { main(); }
```

permissionless

Classical formal logic

NO 

“Show me your permit!”

permissioned

Talk Outline

- Background: whence this work?
- Introduction: grounded arithmetic (GA)
- **Habeas quid: enabling freedom of recursion**
- Metatheory: what properties does GA have?
- Speculation: where might this lead us?

A (new?) reasoning principle

habeas quid

We must *have a thing* in order to use it.

“Python-style dynamic type checking”

Boolean as a “dynamic type”

A term is boolean in grounded deduction *only* if it has been previously *proven* either true or false

$$a\ \mathbf{bool} \equiv a \vee \neg a\ \mathbf{true}$$

New ***habeas quid*** obligations require proving we “have a (boolean) thing” before certain inferences

GA by Example (Paradox)

Habeas quid principle “defuses” paradoxes like:

- The Liar paradox (negation)
- Curry’s paradox (implication)
- Yablo’s paradox (quantification)
- Berry’s paradox (representation)

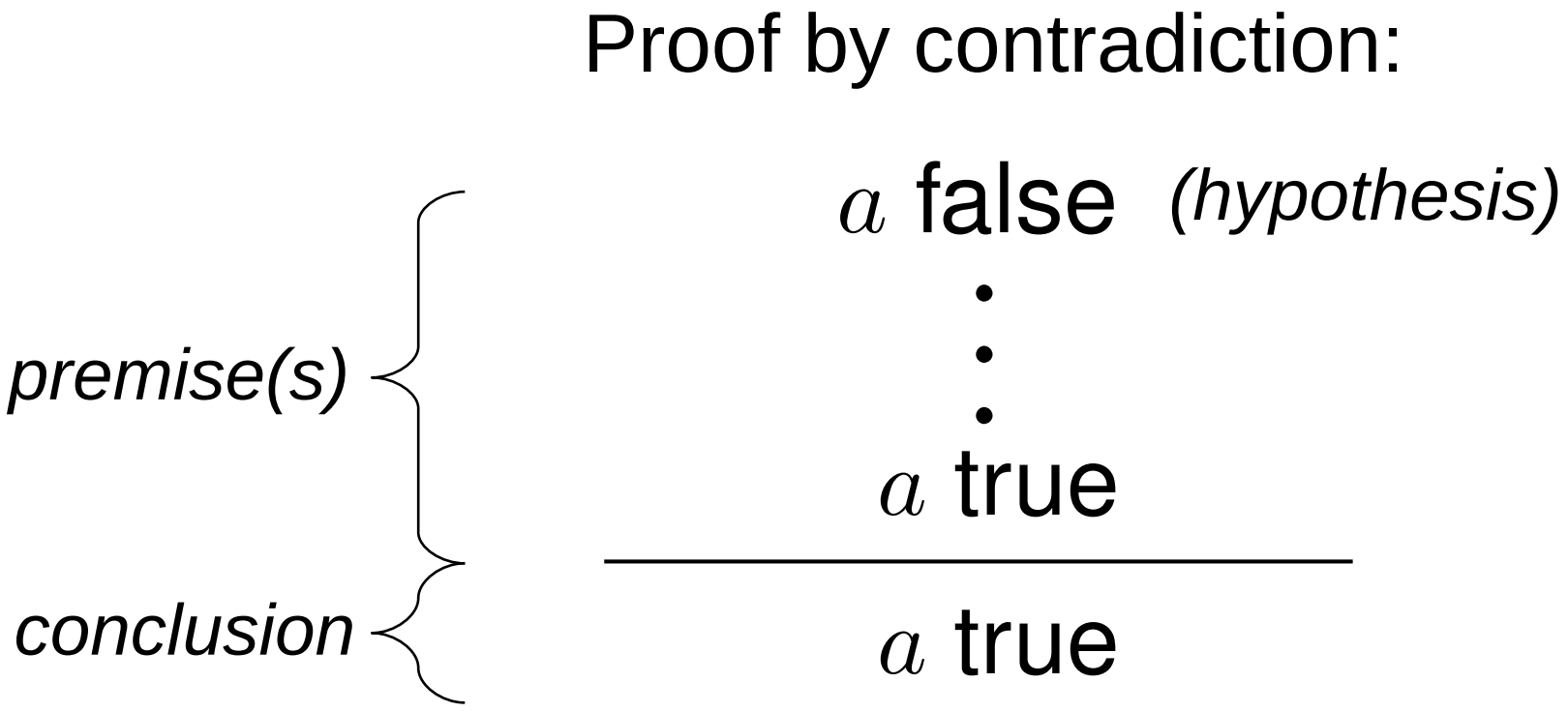
The Liar Paradox

“This sentence is false”

$$L \equiv \neg L$$

Natural deduction – inference rules

Proof by contradiction:



The Liar Paradox in Classical Logic

“This sentence is false” $L \equiv \neg L$

L false (hypothesis)

¬*L* false (*L*’s definition)

L true (negation)

L true (conclusion)

L true (hypothesis)

¬*L* true (*L*’s definition)

L false (negation)

L false (conclusion)

The Liar Paradox in Classical Logic

“This sentence is false” $L \equiv \neg L$

Classical deduction

“Not allowed!”

Recursion must be justified

“Show me your permit!”

Grounded Logical Negation

Let's describe negation in terms of true and false:

$$\frac{a \text{ true}}{\neg a \text{ false}}\,\neg I1 \qquad \frac{a \text{ false}}{\neg a \text{ true}}\,\neg I2 \qquad \frac{\neg a \text{ false}}{a \text{ true}}\,\neg E1 \qquad \frac{\neg a \text{ true}}{a \text{ false}}\,\neg E2$$

Double-negation introduction and elimination hold:

$$\frac{a \text{ true}}{\neg\neg a \text{ true}}\,\neg\neg I \qquad \frac{\neg\neg a \text{ true}}{a \text{ true}}\,\neg\neg E$$

…*unlike* in intuitionistic logic. *But*…

~~Classical~~ proof by contradiction

Grounded

a bool

habeas quid

$$\frac{\begin{array}{c} a \text{ false} \\ \vdots \\ a \text{ true} \end{array}}{a \text{ true}}$$

The Liar Paradox

“This sentence is false” $L \equiv \neg L$

Classical deduction

“Not allowed!”

Recursion must be justified

“Show me your permit!”

Grounded deduction

Valid recursive definition

But is *L* a (bool) thing?

- To use contradiction, *first* prove *L* boolean

Grounded logical implication

GA defines implication as in classical logic:

$$a \to b \equiv \neg a \vee b$$

…again *unlike* intuitionistic logic

And similarly biconditional (if-and-only-if):

$$a \leftrightarrow b \equiv (a \to b) \wedge (b \to a)$$

Grounded logical implication

Introduction and elimination rules for implication:

a bool

habeas quid

$$\frac{\begin{array}{c}a \text{ true} \\ \vdots \\ b \text{ true}\end{array}}{a \to b \text{ true}} \to I$$

(modus ponens)

$$\frac{a \to b \text{ true} \qquad a \text{ true}}{b \text{ true}} \to E$$

Curry's Paradox

"If this sentence is true then pigs fly" $C \equiv C \rightarrow P$

Classical/intuitionistic	**Grounded deduction**
Illegal circular definition!	Valid recursive definition
If allowed, inconsistency	But is *C* boolean?
• With only $\rightarrow I$ and $\rightarrow E$ (no classical LEM)	• To introduce $C \rightarrow P$, *first* prove *C* boolean

Yablo's paradox "without self-reference"

Suppose we have an infinite series of statements:

1) All the following statements are false.
2) All the following statements are false.
3) …

Are any of these statements true (or false)?

Yablo's paradox

Directly expressible in GA using quantifiers:

$$Y_i \equiv \forall_{j>i}(\neg Y_j)$$

Taking indexed Y_i as function $Y(i)$ with argument i

All are ungrounded/nonterminating: Y_i denotes $\perp$

Berry paradox

What (natural) number does this phrase denote?

“the least number not nameable in under ten words”

We can formalize this phrase in GA via reflection:
a computation that *tries to find* such a number,
but on the way recursively re-invokes itself (⊥)

Can we do *useful* reasoning in GA?

Yes.

- Simple example: “even” function (next)
- Mutual recursion example: Cantor pairing
- In-progress: GA in the Isabelle proof assistant
- Metatheory: any general-recursive computation

How useful? Only time, experience will tell…

Satisfying *habeas quid* obligations

Example: $\mathsf{even}(a) \equiv \begin{cases} \mathsf{true} & \mid a = 0 \\ \neg\mathsf{even}(a-1) & \mid a > 0 \end{cases}$

Habeas quid proof by induction:

$$\dfrac{\dfrac{(\mathsf{true})\ \mathsf{bool} \qquad \neg\mathsf{even}(x)\ \mathsf{bool}}{\mathsf{even}(0)\ \mathsf{bool} \qquad \mathsf{even}(x+1)\ \mathsf{bool}}\ \underbrace{\begin{array}{c} x\ \mathsf{nat} \qquad \mathsf{even}(x)\ \mathsf{bool} \\ \vdots \end{array}}}{\forall x\ \mathsf{even}(x)\ \mathsf{bool}}$$

More first-hand use experience

Formalizing Grounded Arithmetic atop Isabelle/Pure

Sascha Kehrli
B.Sc. thesis advised by Bryan Ford and Roger Wattenhofer
September 30, 2025

Abstract:

This thesis presents a foundational formalization of Grounded Arithmetic (GA), a first-order arithmetic based on the principles of Grounded Deduction (GD), directly within the Isabelle/Pure framework. Unlike classical and constructive logics, which impose strict termination requirements on definitions to preserve consistency, GD admits arbitrary recursion at the definitional level. To remain consistent, GA weakens other inference rules, many of which demand explicit *habeas quid* termination proofs of subexpressions as premises. The goal of this thesis is to investigate the feasibility of GA as a practical basis for mathematical and computational reasoning by fully axiomatizing it in Pure.

Untangling mutual recursion

Classical and intuitionistic formal systems: mutually-recursive data types, functions definable only *all at once* with a mutual termination proof

Grounded arithmetic: no such limitation; “freedom of mutual recursion”

Example: Isabelle/HOL (Classical)

Define mutually-recursive types, funs all at once

```
datatype tm = ... | tmIf fm tm tm | ...
and        fm = ... | fmEq tm tm | ....
```

Prove inductive theorems about them all at once

```
lemma to_trm_inj:
  shows "tm_to_trm a = tm_to_trm b ⟹ a = b"
    and "fm_to_trm p = fm_to_trm q ⟹ p = q"
proof (induction a and p arbitrary: b and q)
  ...
```

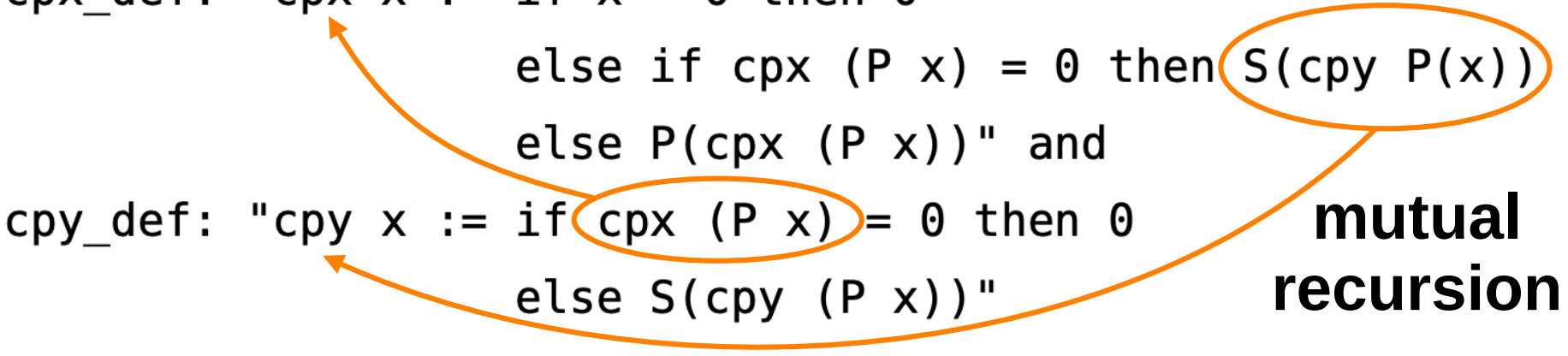
Example: Cantor pairings in GA

Defining pairing and projection recursively:

```
cpair_def: "cpair x y := if y = 0 then div (x * S(x)) 2
                         else cpair x P(y) + x + y + 2"

cpx_def: "cpx x := if x = 0 then 0
                   else if cpx (P x) = 0 then S(cpy P(x))
                   else P(cpx (P x))" and

cpy_def: "cpy x := if cpx (P x) = 0 then 0
                   else S(cpy (P x))"
```



mutual recursion

Kehrli, “Formalizing Grounded Arithmetic atop Isabelle/Pure”

We *can* have “freedom of recursion”

Unconstrained (even “paradoxical”) recursion

- Prove termination separately, later/elsewhere, only for inputs you care about – or not at all

Sane non-local mutual recursion

- Mutually recursive objects need not be defined, or proven terminating, “all at once” together

Talk Outline

- Background: whence this work?
- Introduction: grounded arithmetic (GA)
- Habeas quid: enabling freedom of recursion
- **Metatheory: what properties does GA have?**
- Speculation: where might this lead us?

Grounded arithmetic metatheory

Have a thing? Reasoning around recursion with dynamic typing in grounded arithmetic

Elliot Bobrow, Bryan Ford, and Stefan Milenković

arXiv preprint 2510.25369
October 29, 2025 (first version)

Abstract:

Neither the classical nor intuitionistic logic traditions are perfectly-aligned with the purpose of reasoning about computation, in that neither tradition can permit unconstrained recursive definitions without inconsistency: recursive logical definitions must normally be proven terminating before admission and use. We introduce grounded arithmetic or GA, a formal-reasoning foundation allowing direct expression of arbitrary recursive definitions. GA adjusts traditional inference rules so that terms that express nonterminating computations harmlessly denote no semantic value (i.e., $\perp$) instead of yielding inconsistency. Recursive functions may be proven terminating in GA essentially by "dynamically typing" terms, or equivalently, symbolically reverse-executing the computations they denote via GA's inference rules. Once recursive functions have been proven terminating, logical reasoning about their results reduce to the familiar classical rules. A mechanically-checked formal development of basic grounded arithmetic or BGA – a minimal kernel for GA – shows that BGA simultaneously exhibits the useful metalogical properties of being (a) semantically and syntactically consistent, (b) semantically complete, and (c) sufficiently powerful to express and prove the termination of arbitrary closed Turing-complete computations. This combination is impossible in classical logic due to Göel's incompleteness theorems, but our results do not contradict Gödel's theorems because BGA is paracomplete, not classical. Leveraging BGA's power of computation and reflection, we find that grounded logical operators including quantifiers are definable as non-primitive computations in BGA, despite not being included as primitives.

Basic Grounded Arithmetic (BGA)

Minimalistic computational foundation for GA

- Term syntax: 0, successor, predecessor, if-zero-else, 2-argument function call

$$t \equiv v \mid 0 \mid \mathbf{S}\ t \mid \mathbf{P}\ t \mid t\ 0?\ t : t \mid d(t, t)$$

- Formula syntax: equal (=), unequal (≠)

$$f \equiv t = t \mid t \neq t$$

- Fixed list of 2-argument recursive definitions

Mechanically-verified metatheory

Basic Grounded Arithmetic (BGA)

↓

Logic syntax, proof theory, model theory, recursion theory

↓

Metalogic: Isabelle/HOL (Higher-Order Logic)

↓

Zermelo-Fraenkel set theory with Choice (ZFC)

Inference/proof rules of BGA

(Just) sufficient to prove results of computations

$$\frac{s(\vec{x}) \equiv d\langle\vec{x}\rangle \qquad p\langle d\langle\vec{a}\rangle,\ldots\rangle}{p\langle s(\vec{a}),\ldots\rangle}\equiv I \qquad \frac{a=b}{b=a}=S \qquad \frac{a=b \qquad p\langle a,\ldots\rangle}{p\langle b,\ldots\rangle}=E \qquad \frac{a\neq b}{b\neq a}\neq S \qquad \frac{}{0\ \mathsf{N}}\,0I \qquad \frac{a=b}{\mathrm{S}(a)=\mathrm{S}(b)}\,\mathrm{S}{=}I$$

$$\frac{a\neq b}{\mathrm{S}(a)\neq\mathrm{S}(b)}\,\mathrm{S}{\neq}I \qquad \frac{a\ \mathsf{N}}{\mathrm{S}(a)\neq 0}\,\mathrm{S}{\neq}0I \qquad \frac{a=\mathrm{S}(b)}{\mathrm{P}(a)=b}\,\mathrm{P}I \qquad \frac{c=0 \qquad a\ \mathsf{N}}{(c\ 0?\ a:b)=a}\,?I1 \qquad \frac{c\neq 0 \qquad b\ \mathsf{N}}{(c\ 0?\ a:b)=b}\,?I2$$

No primitive induction! (like Robinson arithmetic)

Semantic model of BGA

Used an operational semantic reduction model…

Term reductions

$$\frac{A(\mathbf{v}_j) = n}{\mathbf{v}_j \Downarrow n} \qquad \frac{}{0 \Downarrow 0} \qquad \frac{a \Downarrow n}{\mathrm{S}(a) \Downarrow n+1} \qquad \frac{a \Downarrow n+1}{\mathrm{P}(a) \Downarrow n} \qquad \frac{c \Downarrow 0 \qquad a \Downarrow n}{c\ 0?\ a : b \Downarrow n} \qquad \frac{c \Downarrow 1+m \qquad b \Downarrow n}{c\ 0?\ a : b \Downarrow n} \qquad \frac{a_0 \Downarrow n_0 \qquad a_1 \Downarrow n_1 \qquad D_i\langle \underline{n_0}, \underline{n_1} \rangle \Downarrow m}{\mathbf{d}_i(a_0, a_1) \Downarrow m}$$

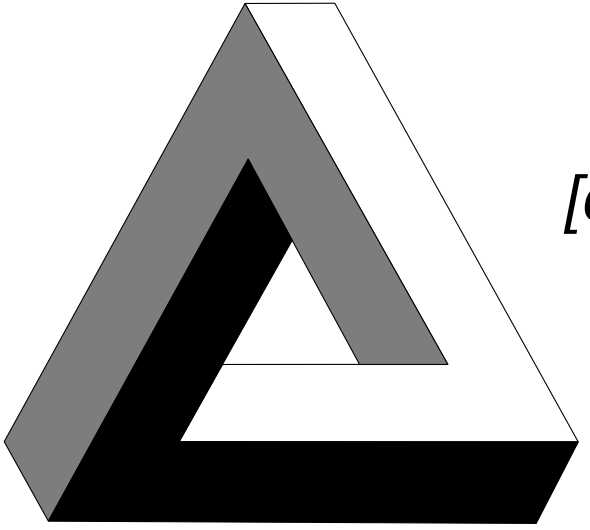
Formula reductions

$$\frac{a \Downarrow n \qquad b \Downarrow n}{a = b \Downarrow 1} \qquad \frac{a \Downarrow n \qquad b \Downarrow m \qquad n \neq m}{a = b \Downarrow 0} \qquad \frac{a \Downarrow n \qquad b \Downarrow n}{a \neq b \Downarrow 0} \qquad \frac{a \Downarrow n \qquad b \Downarrow m \qquad n \neq m}{a \neq b \Downarrow 1}$$

Denotational semantic model should also work

“Gödel’s trilemma”

Expressive Power
arithmetic + computation



In any
[classical or intuitionistic]
formal system,
choose two!

Consistency
“anything provable is true”

Completeness
“anything true is provable”

In Basic Grounded Arithmetic (BGA)

~35K line Isabelle/HOL metatheory shows:

- BGA has **power to express any computation**
  - Via unconstrained recursive definitions
- BGA is semantically & syntactically **consistent**
  - With respect to an operational-semantic model
- BGA is semantically **complete**
  - Anything true in its semantic model is provable

Semantic and syntactic consistency

LEMMA 4.5 (TRUTH PRESERVATION). *The inference rules of BGA in Table 2c are truth preserving.*

THEOREM 4.6 (SEMANTIC CONSISTENCY). *Every theorem of BGA is true in the operational-semantic model of BGA.*

THEOREM 4.7 (SYNTACTIC CONSISTENCY). *There are no terms* t_1, t_2 *such that both '*$t_1 = t_2$*' and '*$t_1 \neq t_2$*' are theorems of BGA.*

Semantic completeness

LEMMA 4.8. *For any well-formed term t that reduces to a natural number r in s steps under all assignments via BGA's operational semantics, the judgment '$\emptyset \vdash t = \underline{r}$' is provable in BGA.*

THEOREM 4.9 (SEMANTIC COMPLETENESS). *Every semantically true formula in BGA is provable using the rules in Table 2c.*

What about syntactic completeness?

Standard (classical) definition:
for every formula *f*, either *f* or ¬*f* is provable.

Impossible in *any* consistent paracomplete system

But unlike in classical logic,
syntactic and semantic completeness *not* aligned

Representation of computation

THEOREM 4.12 (GENERAL RECURSIVE FUNCTIONS). *For any 1-argument general-recursive function $f(x)$ defined by primitive-recursive step function $f_s(x, y)$, there is a definition list D yielding a BGA instance B_D that represents $f(x)$.*

THEOREM 4.14 (UNIVERSAL RECURSION). *The universal BGA instance B_U represents every general-recursive function.*

Why is this combination important?

Intuitive essence:

- Consistency: true and false are distinguishable
- Completeness: truth and proofs well-aligned
  - Can “traverse” freely between truth and provability
- Computation: evidence of power, generality

Together: enable strong Gödel-style reflection

What about logical connectives?

Not primitive in BGA, but definable computations

THEOREM 4.15 (ARITHMETIZATION). *Functions defining the syntax and proof rules of BGA are represented in BGA instance* B_U.

…using Gödel's arithmetic reflection techniques

What about logical connectives?

Not primitive in BGA, but definable computations

THEOREM 4.15 (ARITHMETIZATION). *Functions defining the syntax and proof rules of BGA are represented in BGA instance* B_U.

THEOREM 4.16 (GROUNDED NEGATION). *For any BGA for-*

THEOREM 4.17 (GROUNDED DISJUNCTION). *For any BGA formulas* f_1 *and* f_2, *a formula '*$f_1 \vee f_2$*' is constructible in BGA instance* B_U *such that '*$f_1 \vee f_2$*' is true iff either* f_1 *or* f_2 *are true, and such that '*$f_1 \vee f_2$*' is false iff both* f_1 *and* f_2 *are false.*

What about quantification?

Unlike in classical Peano arithmetic or set theory, quantifiers in grounded arithmetic are computable

…and can even represent mathematical induction

THEOREM 4.18 (INDUCTIVE UNIVERSAL QUANTIFICATION). *For any BGA predicate $p\langle x\rangle$ with one free variable x, a formula '$\forall x\ p\langle x\rangle$' is constructible in BGA instance B_U that: (a) evaluates to true if there are BGA proofs of '$\emptyset \vdash p\langle 0\rangle$' and '$\{x\ \mathbf{N}, p\langle x\rangle\} \vdash p\langle \mathbf{S}\, x\rangle$', and (b) evaluates to false if there is a BGA ~~proof~~ refutation of '$\emptyset \vdash p\langle \underline{n}\rangle$' for some natural number n.*

Upshot: predicate logic “for free”

All logical connectives definable as computations
…using Gödel-style reflection “constructively”

THEOREM 4.20 (GROUNDED PREDICATE LOGIC). *All of the grounded predicate logic operators in Section 3 are represented as computations in the universal BGA instance* B_U.

Summary: “nice things” we *can* have

Practical:

1. Unconstrained recursion
2. Sane mutual recursion

Metalogical:

1. Power + Consistency + Completeness
2. Reflection

…with grounded but not classical reasoning

Talk Outline

- Background: whence this work?
- Introduction: grounded arithmetic (GA)
- Habeas quid: enabling freedom of recursion
- Metatheory: what properties does GA have?
- **Speculation: where might this lead us?**

Work in progress: *more speculative*

Practical usability development

- Beyond arithmetic: rich data types, real, sets
- Can we have “point-like” grounded reals?

Further metatheory development

- How far does power of grounded reasoning go?
- How does it compare with PA, ZF set theory?

Can we learn to “think grounded”?

Personal experience (after several years): Yes.

Even if you believe me…it’s not quick or easy.

And one data-point does not constitute a trend.

Can we learn to “think grounded”?

Not easy, but…

One natural language, ***Aymara***, has built-in notion of *true*, *false*, or *indeterminate* (neither, unknown)

Existence proof that human reasoning is *flexible*

Reflection à la Gödel

Let us try to prove Gödel's theorems about GA.

- Gödel coding and reflection on provability: YES
- Diagonalization (aka fixed-point) lemma: NO
  - Must weaken with *habeas quid* precondition

What does this mean for Gödel's theorems?

Reflection à la Gödel

Gödel 1st "there is an undecidable sentence"

- **YES**: not Gödel's way, but more trivially via Liar

Godel 2nd "a system can't prove itself consistent"

- **NO**: proof doesn't work, no apparent fix
  - For same reason Liar paradox is defused

Bug or feature? *Can GA prove itself consistent?*

Strong “Not True” revisited

We might like to be able to say, and prove, that “the Liar sentence *L* is Not True (and Not False)”

- Without slipping down the slope of revenge

Did Gödel already show us a solution? *Reflection*

- “*L* is Not True” ≡ ¬**Prv**(⌜*L*⌝)
- “*L* is Not False” ≡ ¬**Prv**(⌜¬*L*⌝)
- “*L* is Ungrounded” ≡ ¬**Prv**(⌜*L*⌝) ∧ ¬**Prv**(⌜¬*L*⌝)

Paradoxes and paradoxes-in-hiding

What if we fully “drink the grounded kool-aid” and view classical results from grounded perspective?

“Interesting things” happen to:

- Halting problem
- Cantor’s theorem
- Banach-Tarski paradox

The Halting Problem

Under a reasonable grounded perspective on what constitutes a computational “problem”…

- Must have a well-defined *target* or goal-posts

Search Problem **Sorting Problem** **Halting Problem**

The Halting Problem

Under a reasonable grounded perspective on what constitutes a computational “problem”…

- Must have a well-defined *target* or goal-posts
- Halting problem *statement* is non-terminating!
  - Classical reasoning: pretends it is well-defined
  - Grounded reasoning: no problem to be solved – *habeas quid* barrier to proving it is even a “problem”

Cantor’s “paradox in hiding”

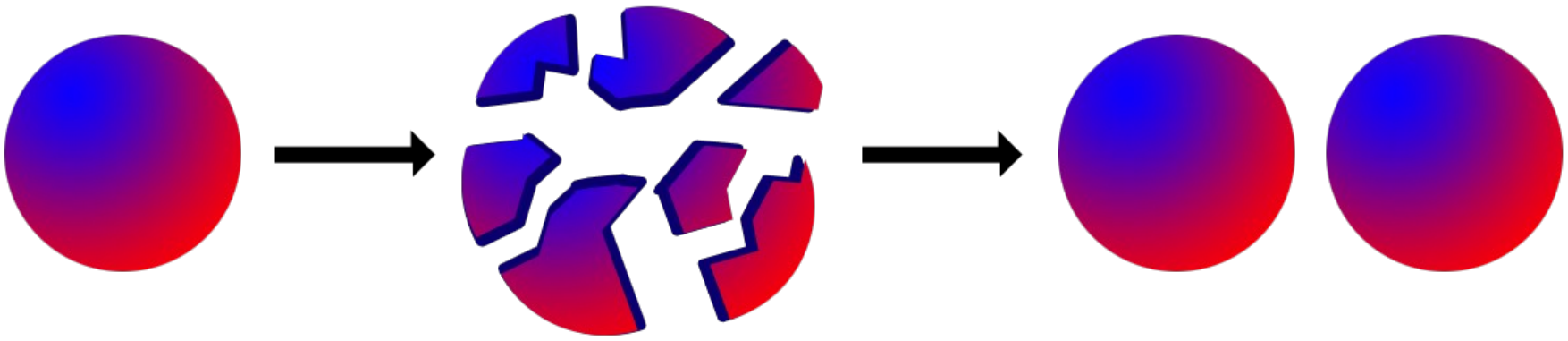
Supposing real numbers have an enumeration *E*, construct a new real number *C* not listed in *E*

- Grounded: reals must “decide” their “position”
  - Cantor’s real “searches for itself” among all reals
  - “Finds itself” only given verifiable *proof C* is a real
  - *C* re-invokes itself: *habeas quid* violation, no proof

Grounded reals enumerable; cardinals collapse

Placing the blame for Banach-Tarski

In ZFC, we can reassemble 1 unit ball into 2 balls



[image credit: cognitive coitus]

Placing the blame for Banach-Tarski

In ZFC, we can reassemble 1 unit ball into 2 balls

Proof depends on Axiom of Choice (AC)

- Triggered intense controversy over AC

But: proof *also* depends on Cantor's Theorem

- Uncountable set ⧹ countable set ≠ empty

Was the "wrong suspect" historically on trial?

Logic vs logic “critiques”

Classical vs grounded:

“Impoverished!”

- Weakened inference
- Excluded middle lost
- Cantor’s paradise lost
- Non-computable lost

Grounded vs classical:

“Hallucinatory!”

- Weakened recursion
- Nonsense as “truth”
- Conjures phantasms
- Reasoning run amok

Conclusion: further reading

“Reasoning around paradox with grounded deduction”

“Have a thing?
Reasoning around recursion…”

“Formalizing grounded arithmetic atop Isabelle/Pure”