Research Article

What Comes First? Mathematical Objects or Their Logical Justification? A Type-Theoretic Perspective for Mathematics Education

Reinhard Oldenburg[1]

1. Universität Augsburg, Germany

Mathematics education has studied how learners come to conceive processes as mathematical objects. Mainstream logical semantics, by contrast, normally starts after such objects are already available: in standard model-theoretic semantics, quantified statements are evaluated relative to a given domain. This essay explores the tension between these two perspectives. It asks whether type theory offers a useful formal vocabulary for describing the constitution of mathematical domains prior to reasoning about them. Inductive types provide a particularly transparent example: the natural numbers are specified through rules for forming and using their elements rather than by first postulating a completed domain. Yet induction is not general enough to account for all mathematical object formation. Herbrand universes and the contrast between the real algebraic numbers and the real numbers illustrate both the power and the limitations of term-generated domains. Dependent type theory suggests a broader perspective in which new domains arise through type formation—including function types, dependent types, inductive types, and suitable quotient constructions. The proposal is not that type theory is a psychological theory of learning, but that it may provide a normative formal counterpart to didactical theories of object constitution such as Sfard's process–object duality and APOS theory.

Corresponding author: Reinhard Oldenburg, reinhard.oldenburg@math.uni-augsburg.de

 doi.org/10.32388/H1VRJZ

1. Introduction

What comes first in mathematics: the objects, or the reasons we give for statements about them? For a practising mathematician this question may appear artificial. In mathematics education, however, the distinction matters. A learner cannot meaningfully justify a statement about functions, limits, quotient groups, or real numbers if the relevant objects are not yet available as objects of thought. The constitution of a mathematical object is therefore not merely a preliminary technicality. It is itself part of learning mathematics.

This point is familiar in mathematics education. [1] distinguishes operational conceptions of mathematical notions from structural ones: a mathematical notion such as number or function may initially be experienced as a process and only later turned into an object. Her account of interiorization, condensation, and reification gives special importance to the difficult transition by which a process becomes something that can itself be acted upon. APOS theory offers a related vocabulary. Actions are interiorized into processes, processes can be encapsulated as objects, and objects and processes are coordinated within larger schemas [2]. Both approaches therefore treat the availability of mathematical objects as an achievement rather than as a starting point. A recent proposal by [3] connects such reification theories with the Curry–Howard correspondence and typed functional computation, suggesting that computational processes can be encapsulated as mathematical objects. The present essay is close to that proposal in spirit, but asks a different foundational question: not primarily how proofs, programs, and processes correspond, but how a *domain of objects* becomes available before quantified reasoning about it can begin.

The standard logical picture begins elsewhere. In model-theoretic semantics, a first-order language is interpreted in a structure with a given domain. The statement $\forall x\, A(x)$ is true in a structure if $A(x)$ holds under every assignment of an element of that domain to x. The semantics is clear and powerful, but from a didactical point of view it starts rather late: the objects over which quantification ranges have already been established.

This observation is not intended as an objection to model theory. Tarskian semantics answers a semantic question: under what conditions is a formula true in a structure? It is not designed to answer a genetic or psychological question: how does a domain of mathematical objects become available to a learner? The two questions belong to different projects. Nevertheless, the contrast raises a useful foundational

question for mathematics education. Is there a logical framework in which the formation of a domain is represented inside the formal system rather than presupposed by the semantics?

Type theory is interesting in this respect. In Martin-Löf's presentation, judgements such as $A\ \mathrm{type}$ and $a : A$ are governed by rules that explain what is required to recognize a type and its elements [4]. For inductive types, the connection to object formation is especially visible. The natural numbers are introduced by constructors for zero and successor together with an elimination or induction principle. One can therefore read the formal order as

$$\text{rules of construction} \longrightarrow \text{objects} \longrightarrow \text{reasoning about those objects.}$$

This resembles, at least formally, the didactical movement from activity or process toward object.

The resemblance should not be exaggerated. A formal type theory is not a cognitive model, and a child's construction of number is not adequately described as the execution of the introduction rules for $\mathbb{N}$. The claim explored here is weaker: type theory may offer a useful *normative reconstruction* of different ways in which mathematical domains become available for reasoning. Inductive generation will turn out to be only one such way. The discussion of the contrast between $\mathbb{R}_{\mathrm{alg}}$ and $\mathbb{R}$ will show why function formation, completion, and quotient constructions are equally important. The resulting perspective suggests that learning mathematics may be described not only as acquiring propositions and proofs within a fixed universe, but also as acquiring new kinds of objects over which propositions can subsequently be formed and justified.

2. The Mainstream View: Model-Theoretic Semantics and Its Need for Objects

A standard model-theoretic semantics for first-order logic begins with a language L and an L-structure $\mathscr{M}$ see, e.g., [5]. The structure contains a non-empty domain $|\mathscr{M}|$ together with interpretations of the non-logical symbols of L. Truth is then defined recursively relative to variable assignments. In particular,

$$\mathscr{M}, s \models \forall x\, A(x)$$

means that for every $d \in |\mathscr{M}|$, the formula $A(x)$ is true under the assignment that sends x to d [6].

For logic, this separation between syntax and interpretation is one of the strengths of the framework. The same formula can be studied in many structures, and logical validity can be characterized as truth in

all appropriate structures. The framework also makes clear that quantification is always quantification over a domain supplied by an interpretation. In this sense, the objects are semantically prior to the truth conditions of quantified statements.

For mathematics education this order can be misleading if it is taken as an implicit model of mathematical understanding. Consider a student first encountering functions. To evaluate a statement such as “every continuous function on a compact interval is bounded”, functions must have become available to the student as objects about which quantification is meaningful. The same issue becomes sharper with sequences of functions, quotient spaces, equivalence classes, ideals, or random variables. In advanced mathematics the objects quantified over are often themselves the result of earlier acts of construction and reification.

One could reply that model theory has no obligation to represent this development, and this is correct. The present point is therefore not that model theory lacks something required for its own purposes. Rather, it is that its semantic architecture distinguishes poorly between two educational situations that are very different. In one situation the learner already possesses a domain and learns to prove new statements about its objects. In another, the central learning achievement is precisely the formation of a new domain whose elements can subsequently enter statements and proofs.

Set theory can be used for such constructions. Because functions can be realized as certain subsets of Cartesian products, quantification over them is just first-order quantification over sets. Avoiding higher-order logic is beneficial both from a theoretical and from a computational point of view, but for learners it is a challenge to see functions, sequences, real numbers, and algebraic structures all represented as sets: very different forms of object constitution are encoded by the same membership relation. Type theory, by contrast, can make the formation of different kinds of objects visible in the formal language itself. In Martin-Löf type theory, judgements of the forms $A\,\mathrm{type}$ and $a : A$ are governed by formation, introduction, elimination, and equality rules whose role is explicitly connected to meaning explanations [4]. In the Calculus of Constructions, dependent products make it possible for types to depend on terms and for propositions and proofs to be represented within a single typed calculus [7]. The relevant contrast is therefore not that type theory avoids semantics, but that important acts of domain formation have an explicit proof-theoretic representation rather than occurring only in an external set-theoretic coding.

3. Inductive Types: The Case of the Natural Numbers

The natural numbers provide the simplest case in which formal construction and a didactical story of object formation can be placed side by side. The familiar Peano-style characterization already presents $\mathbb{N}$ through a distinguished starting point 0, a successor operation S, conditions separating 0 from successors and making successor injective, and an induction principle [8]. In first-order Peano arithmetic, induction is an axiom *schema*: for each formula $P(n)$ one has an axiom of the form

$$\big(P(0) \wedge \forall n\,(P(n) \to P(Sn))\big) \to \forall n\,P(n).$$

This point is conceptually important. Mathematical induction is not a general logical law on a par with modus ponens or universal instantiation; it belongs to the non-logical theory of the natural numbers. It expresses a specific structural feature of this domain: starting from 0 and passing to successors is sufficient for reasoning about every natural number.

This connects naturally with counting. In early number learning, children do not normally encounter $\mathbb{N}$ first as a completed set. Number is coordinated with ordered counting activity, the production of a stable sequence of number words, one-to-one correspondence, and the cardinal significance of the last word uttered [9]. The formal successor operation is of course not a psychological theory of counting, but the operation “one more” gives a didactically accessible interpretation of the generative pattern

$$0, \quad S(0), \quad S(S(0)), \quad \ldots.$$

The natural-number concept is thereby connected to a repeatable process before it is treated as a completed structure.

An inductive type makes this generative reading internal to the formal system. Schematically, one starts by introducing a type name $\mathbb{N}$ for the new type of objects, a constant 0 of that type, and the deduction rule that if n is of that type, then so is $S(n)$. Hence, S can be interpreted as the successor function (but it acquires this meaning only through these rules).

$$\mathbb{N}\ \text{type}, \qquad 0 : \mathbb{N}, \qquad \frac{n : \mathbb{N}}{S(n) : \mathbb{N}}.$$

But constructors alone are not enough. The crucial rule is the eliminator. In dependent form, if a family of propositional types $P(n)$ is available, together with

$$p_0 : P(0) \qquad \text{and} \qquad p_S : \prod_{n:\mathbb{N}} \big(P(n) \to P(Sn)\big),$$

then one obtains

$$\mathrm{ind}_{\mathbb{N}}(p_0, p_S) : \prod_{n:\mathbb{N}} P(n).$$

Thus the induction principle is not appended as an unrelated axiom after $\mathbb{N}$ has been specified. It is part of what governs the use of objects generated by the inductive definition. Martin-Löf's meaning explanations make this epistemic role explicit: to understand a type involves knowing its canonical forms and the rules governing how its elements may be constructed and used [4].

This sharpens the contrast with the model-theoretic presentation. In a model of first-order arithmetic, a domain is supplied and the induction schema is then required to hold for the definable properties expressible in the language. In the inductive type, by contrast, introduction and elimination are coordinated: the same zero/successor pattern that generates canonical numerals also determines the principle by which arbitrary elements of the type may be reasoned about. Informally, the formal order becomes

$$\begin{aligned} \text{generation by zero and successor} &\longrightarrow \text{the domain of natural numbers} \\ &\longrightarrow \text{inductive reasoning over that domain.} \end{aligned}$$

The didactical attraction is clear. Sfard's process–object duality suggests that structural conceptions of number are built upon operational ones [1]; APOS likewise emphasizes the encapsulation of processes into objects [2]. The inductive presentation offers a formal analogue in which a repeatable process of successor formation is inseparable from the structural principle of induction. In this sense, induction is especially revealing: it is simultaneously a way of proving propositions about $\mathbb{N}$ and a statement about what counts as exhausting the domain of $\mathbb{N}$.

Induction can therefore be viewed as a hinge between a *genetic* and an *epistemic* perspective on the natural numbers. Genetically, the zero/successor pattern says how arbitrary numerals can arise from a repeatable "one more" operation. Epistemically, the induction principle says what is sufficient for knowing that a proposition holds of every natural number: establish it at the starting point and show that it is preserved by the same operation that generates the domain. The construction principle and the proof principle thus have the same structural backbone. This does not show that children acquire induction simply by learning to count. It does, however, identify a particularly transparent case in which the constitution of mathematical objects and the justification of universal statements about them are formally coordinated rather than sequentially independent.

The analogy nevertheless has limits. Type-theoretic rules do not describe the empirical trajectory by which children acquire natural number concepts. They abstract away from embodiment, language, cardinality, ordinality, cultural representations, and social interaction. Nor should Peano's axioms themselves be read as a learning theory. Their value here is comparative: they show that even in a classical axiomatization, induction is domain-specific, while type theory goes one step further by integrating the induction principle with the rules that introduce the domain. The natural numbers therefore provide a compelling first case of object constitution by rules. The next question is whether the same pattern scales to mathematical domains whose elements are not generated by repeated application of constructors.

4. Power and Limitations of Inductive Approaches

4.1. Herbrand Universes and Term-Generated Domains

First-order logic itself contains an important construction that resembles an inductive domain: the Herbrand universe. Given constants and function symbols, closed terms are generated recursively. Constants are ground terms; whenever f is an n-ary function symbol and $t_1, \ldots, t_n$ are ground terms, $f(t_1, \ldots, t_n)$ is again a ground term. The resulting universe is syntactic and term-generated. Herbrand's theorem connects such terms with witnessing and propositional instances of first-order formulas and has become fundamental for automated reasoning [10][11].

From the present perspective the Herbrand universe is instructive because it shows how much can be obtained without postulating a completed semantic universe of "existing" objects. Start, for example, with 0, 1, addition, multiplication, and additive inverse. Modulo the usual ring equations, the closed terms name the integers. Add multiplicative inverses, with the necessary conventions, and rational numbers become term-definable. If suitable Skolem functions [11][12] are added to select algebraic witnesses—for instance roots whose existence is guaranteed by axioms for real closed fields—the term-generated subfield of the intended real field can be extended through algebraic operations.

However, two distinctions are essential. First, a Herbrand term is initially a syntactic object, not an element of an intended mathematical structure. Equations may identify many different Herbrand terms. Second, the existence of a Skolem term does not by itself provide a psychologically or computationally transparent construction of the object it denotes. Skolemization supplies names for witnesses required

by existential statements. It therefore captures one important sense of construction—availability through terms—but not every sense relevant to learning.

4.2. From $\mathbb{Q}$ to $\mathbb{R}_{\text{alg}}$

The theory of real closed ordered fields makes the point particularly sharp. Starting from $\mathbb{Q}$ and closing under algebraic operations and the roots required for real closure leads, inside the intended real field, to the field $\mathbb{R}_{\text{alg}}$ of real algebraic numbers. Every element of $\mathbb{R}_{\text{alg}}$ is the root of a non-zero polynomial with rational coefficients, and the field is countable.

Yet $\mathbb{R}_{\text{alg}}$ and $\mathbb{R}$ satisfy exactly the same first-order sentences in the language of ordered fields. Both are real closed fields, and the first-order theory of real closed fields is complete and admits quantifier elimination [13][14]. Indeed, the inclusion $\mathbb{R}_{\text{alg}} \subseteq \mathbb{R}$ is elementary in this language. No first-order sentence formulated solely with the ordered-field operations can say: "the domain is the full uncountable field of real numbers rather than the real algebraic numbers."

This is a useful warning against identifying a theory with a unique intended domain. A first-order axiom system may determine all first-order consequences relevant to a given language without determining the cardinality or the full ontology of its intended model. The difference between $\mathbb{R}_{\text{alg}}$ and $\mathbb{R}$ is mathematically enormous, but invisible to first-order ordered-field sentences.

For a didactical reading, however, the more interesting fact is constructive. $\mathbb{R}_{\text{alg}}$ can be understood as the result of repeatedly adjoining solutions to algebraic problems. Its elements remain connected, in principle, with finite algebraic descriptions. Thus, it may be closer to the mental model of students who incrementally extend their view of the real numbers. The passage to $\mathbb{R}$ requires a different kind of move.

4.3. Why This Still Does Not Give $\mathbb{R}$

In a countable first-order language there are only countably many finite terms. No purely term-generated Herbrand universe of this kind can therefore provide one distinct term for every real number. This simple cardinality observation reflects a deeper conceptual point: the usual construction of $\mathbb{R}$ does not proceed by repeatedly applying finitely many algebraic constructors to previously obtained numbers.

Instead, one completes $\mathbb{Q}$. A real number may be represented by a Cauchy sequence of rationals, or by a Dedekind cut. Consider the Cauchy-sequence route. The relevant objects are functions

$$a : \mathbb{N} \to \mathbb{Q}.$$

One then selects those satisfying a Cauchy condition and identifies sequences whose difference converges to zero. Thus the crucial new domain is not generated merely by adding further number constructors. One first makes *sequences themselves* into objects. In a classical set-theoretic account, the function space $\mathbb{Q}^{\mathbb{N}}$ is already uncountable even though $\mathbb{N}$ and $\mathbb{Q}$ are countable.

This is precisely the kind of reification emphasized in mathematics education. An indefinitely extendable process

$$q_0, q_1, q_2, \ldots$$

becomes a single mathematical object, a sequence; equivalence classes of suitable sequences then become new objects, the reals. The passage from $\mathbb{R}_{\text{alg}}$ to $\mathbb{R}$ therefore reveals a limitation of viewing all mathematical objects as inductively generated. Some are, but others arise when functions, sequences, equivalence classes, completions, or structures are themselves admitted as objects.

The same example clarifies a common logical misunderstanding. Cantor's argument for the uncountability of $\mathbb{R}$ cannot be expressed as a statement quantifying over arbitrary functions $\mathbb{N} \to \mathbb{R}$ in the one-sorted first-order language of ordered fields, because such functions are not elements of the field domain. But Cantor's theorem can certainly be formalized in first-order set theory, where functions are themselves sets and hence first-order objects. The issue is therefore not simply first-order versus higher-order logic. It is also what kinds of objects the chosen formal universe makes available for quantification. The downward Löwenheim–Skolem theorem reinforces this point: first-order theories in countable languages cannot in general force all their infinite models to have one prescribed infinite cardinality [14].

For mathematics education this suggests a shift of emphasis. The significant step in understanding Cantor's proof is not only learning a new proposition about $\mathbb{R}$. It is being able to treat functions from $\mathbb{N}$ to $\mathbb{R}$ as objects over which one can quantify and on which one can perform a diagonal construction. The hierarchy of objects matters to the proof.

5. Type Theory: Object Constitution by Type Formation

The preceding examples suggest that inductive generation is too narrow to provide a general account of mathematical object constitution. Type theory offers a broader idea: new domains can be formed from domains already available. The key notion is therefore not induction alone but *type formation*. Two historically related systems help clarify this claim.

5.1. The Calculus of Constructions

The Calculus of Constructions (CoC) of [7] is a higher-order dependent typed λ-calculus designed to serve simultaneously as a language of terms and a logic of propositions and proofs. Its central constructor is the dependent product

$$\prod_{x:A} B(x),$$

which specializes to an ordinary function type $A \to B$ when B does not depend on x. Under the propositions-as-types reading [15], the same constructor also represents universal quantification and implication in the appropriate cases. What matters for the present argument is that dependency is represented internally: once A is available, one can form families $B(x)$ indexed by objects $x : A$ and types of functions whose outputs inhabit those varying types.

This already goes substantially beyond an account in which one merely generates more objects of a fixed kind. Forming a function type changes what counts as an object of discourse. Given A and B, an object $f : A \to B$ is itself available for further quantification and construction. A hierarchy such as

$$A,\ B \quad \rightsquigarrow \quad A \to B \quad \rightsquigarrow \quad (A \to B) \to C$$

therefore represents successive objectifications of operations. From a didactical point of view, this is close to the familiar observation that learners may first execute a procedure and only later treat the procedure itself as an object that can be compared, transformed, or quantified over.

The original CoC, however, does not contain primitive inductive type declarations of the modern form. Natural numbers and other data can be encoded impredicatively, but the calculus does not take a declaration with constructors such as 0 and S and automatically generate the corresponding induction and recursion principles. For the didactical question considered here, this matters: the attractive zero/successor/induction package of Section 3 is not yet the general mechanism of the original CoC.

5.2. The Calculus of Inductive Constructions

The Calculus of Inductive Constructions (CIC) extends this setting with primitive inductive definitions. The early extension by [16] adds inductively defined types to the Calculus of Constructions; later formulations became the formal basis of Coq/Rocq [17]. An inductive declaration does more than add new constants. It specifies constructors subject to well-formedness conditions and supplies associated

elimination or induction principles. Consequently, the formation of a domain and the legitimate modes of reasoning from its constructors are coordinated inside the calculus.

CIC is therefore a better illustration than CoC alone of the thesis developed in this essay. It combines two distinct mechanisms of object constitution. First, dependent type formation allows new domains of functions, families, and structured objects to be built from existing ones. Second, inductive declarations introduce domains generated by specified constructors. The natural numbers use the second mechanism; many domains needed in analysis rely essentially on the first as well.

The reals illustrate the point. Given $\mathbb{N}$ and $\mathbb{Q}$, one can form the function type

$$\mathbb{N} \to \mathbb{Q}.$$

A type of Cauchy sequences can then be represented schematically by

$$\sum_{a:\mathbb{N}\to\mathbb{Q}} \mathrm{isCauchy}(a),$$

and an appropriate quotient or setoid construction identifies equivalent sequences. The exact implementation depends on the type theory; quotient types are not primitive in every version of CIC. The structural lesson is independent of those details. The passage to $\mathbb{R}$ is not another application of a successor-like inductive constructor. It requires the formation of a higher type whose elements are themselves infinite approximation procedures, followed by an identification of representations.

5.3. Contexts as a Didactical Analogy

Dependent type theory also suggests a useful way to describe the changing repertoire of objects available to a learner. Judgements are made relative to a context Γ, which records assumed types and terms. This formal notion should not be identified with a learner's mental state (prior knowledge), but it supplies a suggestive distinction between two kinds of enrichment. One is propositional enrichment: new facts are proved about objects already represented in Γ. The other is conceptual enrichment: a new type becomes available, perhaps together with constructors, operations, or identifications. Schematically,

$$\Gamma \quad \rightsquigarrow \quad \Gamma, A : \mathscr{U} \quad \rightsquigarrow \quad \Gamma, A : \mathscr{U}, a : A, \dots$$

or, when A is constructed from previous material,

$$\Gamma \vdash A \ \mathrm{type}.$$

Only after this enrichment can propositions quantifying over A be formulated internally.

The proposal is attractive because it distinguishes forms of conceptual development that a one-sorted foundational coding tends to flatten. Natural numbers may be introduced inductively. Rational numbers may arise through a quotient or field-of-fractions construction. Real numbers may arise through completion. Functions form a new domain from old ones. Algebraic structures package a carrier type together with operations and laws. These are mathematically different formation acts even when set theory can encode all of them uniformly as sets. Moreover, as noted in [18], unlike sets, types do not presuppose that the totality of objects is already constructed or known.

There is also a connection with current educational interest in interactive theorem provers. Systems such as Lean expose type information, hypotheses, constructors, and proof states explicitly, and a growing literature studies how students interact with such systems in university mathematics [19]. Most of this work understandably concerns proof activity, formalization, and human–computer interaction. The present argument points to a complementary question: could type-theoretic environments also make the *formation of mathematical domains* pedagogically visible? Defining a sequence type, a subtype satisfying a condition, or a quotient is then not merely technical preparation for proof; it may correspond to a mathematically significant transition in what is available as an object of thought.

6. Discussion

The central claim of this essay is modest. Type theory should not be proposed as a psychological theory of mathematical learning. Sfard's theory and APOS are concerned with cognitive development, representations, activity, and the empirical difficulties learners encounter. Type theory is a formal system. No correspondence between a formal introduction rule and a mental construction follows merely from their structural similarity.

Nevertheless, the comparison may be theoretically useful. Mathematics education often studies the transition from process to object, while logic is often introduced to students primarily as a means of expressing and validating propositions. A type-theoretic perspective brings a third component into view: rules governing when a new *kind of object* is available at all. It therefore provides a formal counterpart to the didactical distinction between reasoning within an established domain and constructing the domain itself. The natural numbers make the point especially sharply: induction links the generative structure by which the domain is presented to the justificatory structure by which universal claims over that domain are established. Object constitution and proof are not merely adjacent topics in that case; they are organized by the same zero/successor pattern.

This proposal intersects directly with [3], where I argued that the Curry–Howard correspondence can connect reification, language, proof, and computation, and illustrated process encapsulation with computational representations of mathematical objects, including real numbers. The present account should be read as complementary: Curry–Howard emphasizes the correspondence between proofs and programs and offers a precise model of process/object reification. The argument here emphasizes a different axis: the formation of the *types or domains over which propositions are subsequently formed*. The distinction becomes especially visible in the contrast between $\mathbb{R}_{\text{alg}}$ and $\mathbb{R}$. A proof-theoretic account of propositions alone does not explain why sequences, functions, or equivalence classes have become admissible objects of quantification; type formation makes that change explicit.

Three cases considered above illustrate different modes of object constitution. The natural numbers exemplify generative or inductive construction. The real algebraic numbers exemplify closure under algebraic operations and witnesses. The real numbers exemplify a completion in which infinite processes, represented for instance by Cauchy sequences, are themselves objectified and then identified under an equivalence relation. The progression

$$\mathbb{N} \quad \rightsquigarrow \quad \mathbb{Q} \quad \rightsquigarrow \quad (\mathbb{N} \to \mathbb{Q}) \quad \rightsquigarrow \quad \mathbb{R}$$

is therefore not a single repeated inductive mechanism. It involves qualitatively different acts of mathematical organization.

This may help sharpen a question already present in process–object theories. Reification is often discussed as a general cognitive transition, but different mathematical constructions may demand different kinds of reification. Treating the output of an iteration as an object is not the same as treating an infinite sequence as an object; treating a sequence as an object is not the same as treating an equivalence class of sequences as an object. Type formation offers a formal vocabulary for distinguishing these cases rather than grouping them all under the single category of "object".

The perspective also changes how one might describe proof learning. Proof is not always reasoning about a stable ontology. Some proofs require the learner first to ascend to a new level of objects. Cantor's theorem is an example: the proof requires functions or sequences to be available as objects of quantification. Similarly, many arguments in algebra require homomorphisms, ideals, quotient structures, or spaces of maps to become objects in their own right. Difficulties with proof may therefore sometimes originate upstream, in incomplete object constitution, rather than in inference alone.

These observations suggest empirical questions. When students fail to understand a quantified statement, is the obstacle logical, or is the quantified domain not yet objectified for them? What additional conceptual work is required when passing from elements to functions between elements, or from representatives to equivalence classes? Can the explicit type structure of an interactive theorem prover help diagnose such transitions, or does it merely add formal overhead? These are educational questions, not consequences of type theory, and they require empirical investigation.

A final caution concerns foundations. Type theory does not eliminate semantic presuppositions altogether. Formal systems themselves have metatheories and models, and an abstract type $A : \mathscr{U}$ can be assumed without constructing any of its elements. The relevant advantage is therefore not that type theory magically creates all mathematical objects. Rather, it distinguishes within the object language between assuming a type, constructing a type from previous ones, constructing an element of a type, and proving a proposition about its elements. This explicit stratification is precisely what may make it interesting for mathematics education.

The question posed at the beginning can thus receive a qualified answer. Mathematical objects do not always come before logical justification, nor are they simply produced by it. In mathematical practice and learning, domains of objects and forms of reasoning develop together. A logical framework for mathematics education should therefore be able to represent not only truth and proof but also the formation of the domains over which truth and proof are articulated. Type theory offers one promising formal language for making that dimension visible.

Looking back at the exposition above, it is striking how many concepts and procedures in formal mathematics parallel—or have direct analogues in—notions developed to describe student learning. Just as embodied mathematics [20] posits that formal mathematical structures reflect bodily experiences, this paper suggests that they may also reflect foundational learning processes.

References

1. a, b *Sfard A (1991). "On the Dual Nature of Mathematical Conceptions: Reflections on Processes and Objects a s Different Sides of the Same Coin." Educ Stud Math. 22(1):1–36. doi:10.1007/BF00302715.*
2. a, b *Dubinsky E, McDonald MA (2001). "APOS: A Constructivist Theory of Learning in Undergraduate Mathe matics Education Research." In: Holton D, Artigue M, Kirchgräber U, Hillel J, Niss M, Schoenfeld A, editors. T*

he Teaching and Learning of Mathematics at University Level. New ICMI Study Series; vol. 7. Dordrecht: Kluwer Academic Publishers. p. 275–282. doi:10.1007/0-306-47231-7_25.

3. ^a, ^b*Oldenburg R (2024). "Reification, Curry–Howard Correspondence, and Didactical Consequences." Qeios. Qeios ID S1074V. doi:10.32388/S1074V.*
4. ^a, ^b, ^c*Martin-Löf P (1984). Intuitionistic Type Theory. Naples: Bibliopolis.*
5. ^*Zach R, Open Logic Project (2024). Sets, Logic, Computation: An Open Logic Text. Open Logic Project. https://builds.openlogicproject.org/.*
6. ^*Hodges W (1997). A Shorter Model Theory. Cambridge: Cambridge University Press.*
7. ^a, ^b*Coquand T, Huet G (1988). "The Calculus of Constructions." Inf Comput. **76**(2–3):95–120. doi:10.1016/0890-5401(88)90005-3.*
8. ^*Peano G (1889). Arithmetices principia, nova methodo exposita [Principles of Arithmetic, Presented by a New Method]. Turin: Fratres Bocca.*
9. ^*Gelman R, Gallistel CR (1978). The Child's Understanding of Number. Cambridge, MA: Harvard University Press.*
10. ^*Herbrand J (1930). Recherches sur la théorie de la démonstration [Investigations in Proof Theory]. PhD thesis. Université de Paris.*
11. ^a, ^b*Bundy A (1983). The Computer Modelling of Mathematical Reasoning. London: Academic Press.*
12. ^*Skolem T (1920). "Logisch-kombinatorische Untersuchungen über die Erfüllbarkeit oder Beweisbarkeit mathematischer Sätze nebst einem Theorem über dichte Mengen" [Logical-Combinatorial Investigations on the Satisfiability or Provability of Mathematical Propositions Together with a Theorem on Dense Sets]. Skr Videnskapsselskat Kristiania I Mat Naturvidensk Kl [Writings Published by the Scientific Society in Kristiania. I. Mathematics and Natural Sciences Class]. **1920**(4):1–36.*
13. ^*Tarski A (1951). A Decision Method for Elementary Algebra and Geometry. 2nd ed. Berkeley: University of California Press.*
14. ^a, ^b*Marker D (2002). Model Theory: An Introduction. Graduate Texts in Mathematics; vol. 217. New York: Springer.*
15. ^*Wadler P (2015). "Propositions as Types." Commun ACM. **58**(12):75–84. doi:10.1145/2699407.*
16. ^*Coquand T, Paulin C (1990). "Inductively Defined Types." In: Martin-Löf P, Mints G, editors. COLOG-88. Lecture Notes in Computer Science; vol. 417. Berlin: Springer. p. 50–66. doi:10.1007/3-540-52335-9_47.*
17. ^*The Coq Development Team (2024). "The Coq Proof Assistant Reference Manual." Inria. https://coq.inria.fr.*

18. ^*Oldenburg R (2025). "On the Use of Conceptual Knowledge from Computer Science in Mathematics Education." In: Casabò MB, Carreira S, Bolondi G, Gaidoschick M, Spagnolo C, editors. Proceedings of the Fourteenth Congress of the European Society for Research in Mathematics Education (CERME14), February 3–7, 2025, Bozen-Bolzano, Italy. Bozen-Bolzano: Free University of Bozen-Bolzano and ERME. p. 1846–1853. https://hal.science/hal-05230093v1.*

19. ^*Iannone P, Thoma A (2026). "It Feels Like Sort of Cheating in Some Ways that You Don't Fully Show that You've Understood the Proofs: Mathematics Students Coding Proofs with an Interactive Theorem Prover." Digital Experiences in Mathematics Education. **12**(2):366–392. doi:10.1007/s40751-025-00193-w.*

20. ^*Lakoff G, Núñez RE (2000). Where Mathematics Comes From: How the Embodied Mind Brings Mathematics into Being. New York: Basic Books.*

Declarations

Funding: No specific funding was received for this work.

Potential competing interests: No potential competing interests to declare.