Homotopy Theory and

formalization

~~foundations~~ of

Mathematics

We need to look at the foundations again because of the

Proof correctness problem

Two components:

1. There is an accumulation of results whose proofs the math community can not fully verify

2. There are more and more examples of proofs which have been accepted and later found to be incorrect

This is a much more serious problem for math than it would be for any science because the main strength of mathematics is in its ability to build on multiple layers of previous constructions

There is only one solution other than simply slowing down:

Automated proof verification

Ideally, a paper submitted to a journal should contain text for human readers integrated with references to formalized proofs of all the results. Before being send to a referee the publisher runs all these proofs through a proof checker which verifies their validity. What remains for a referee is to check that the paper is interesting and that the formalizations of the statements correspond to their intended meaning.

Why aren't we there yet?

Attempts to construct "proof assistants" to back to the late 60-ies. Two original ones:

1. Automath - N.G. de Bruijn, late 60-ies

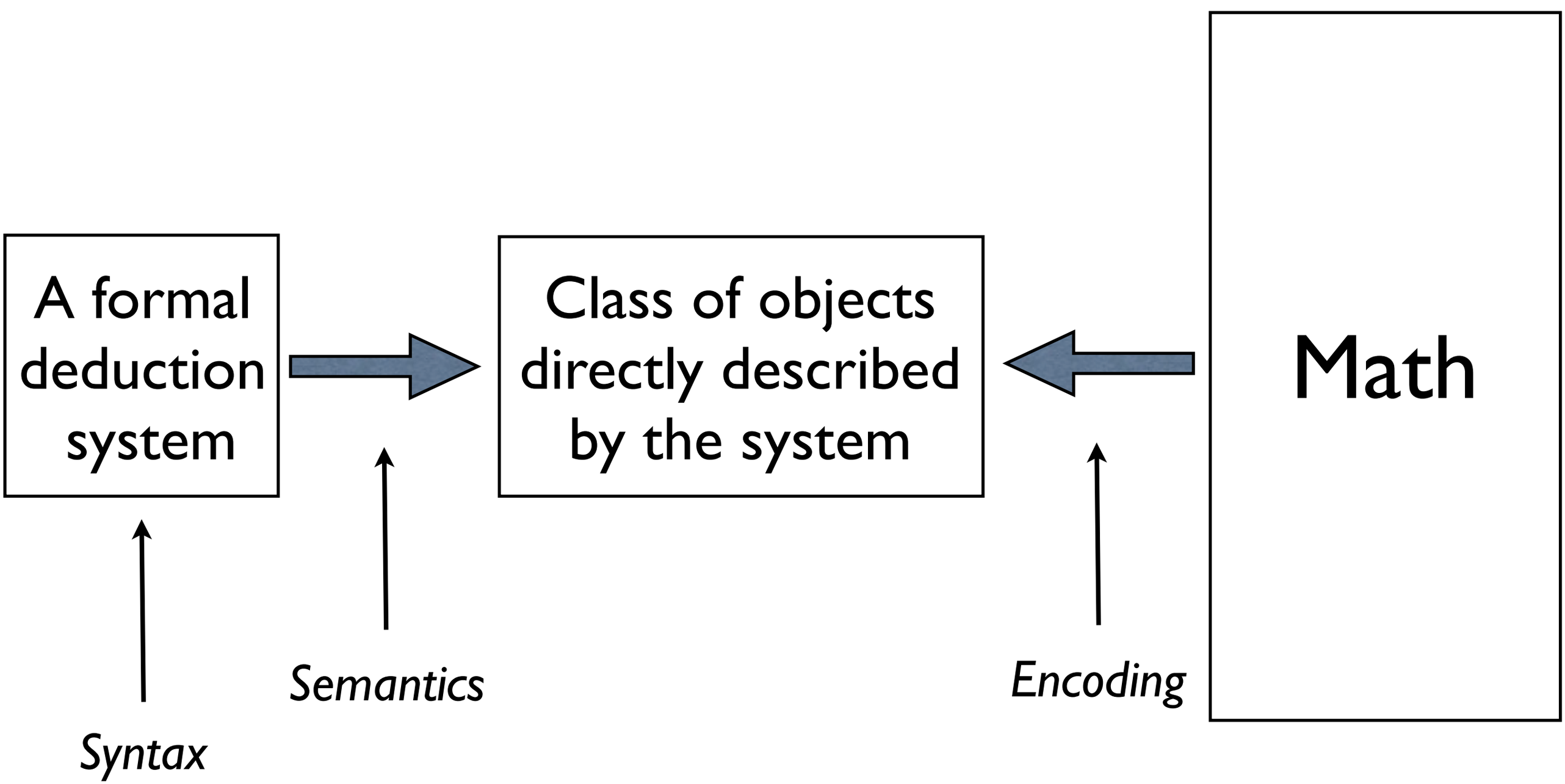
2. Mizar - A. Tryboulec, early 70-ies

From these systems and their numerous descendants we learn main two lessons:

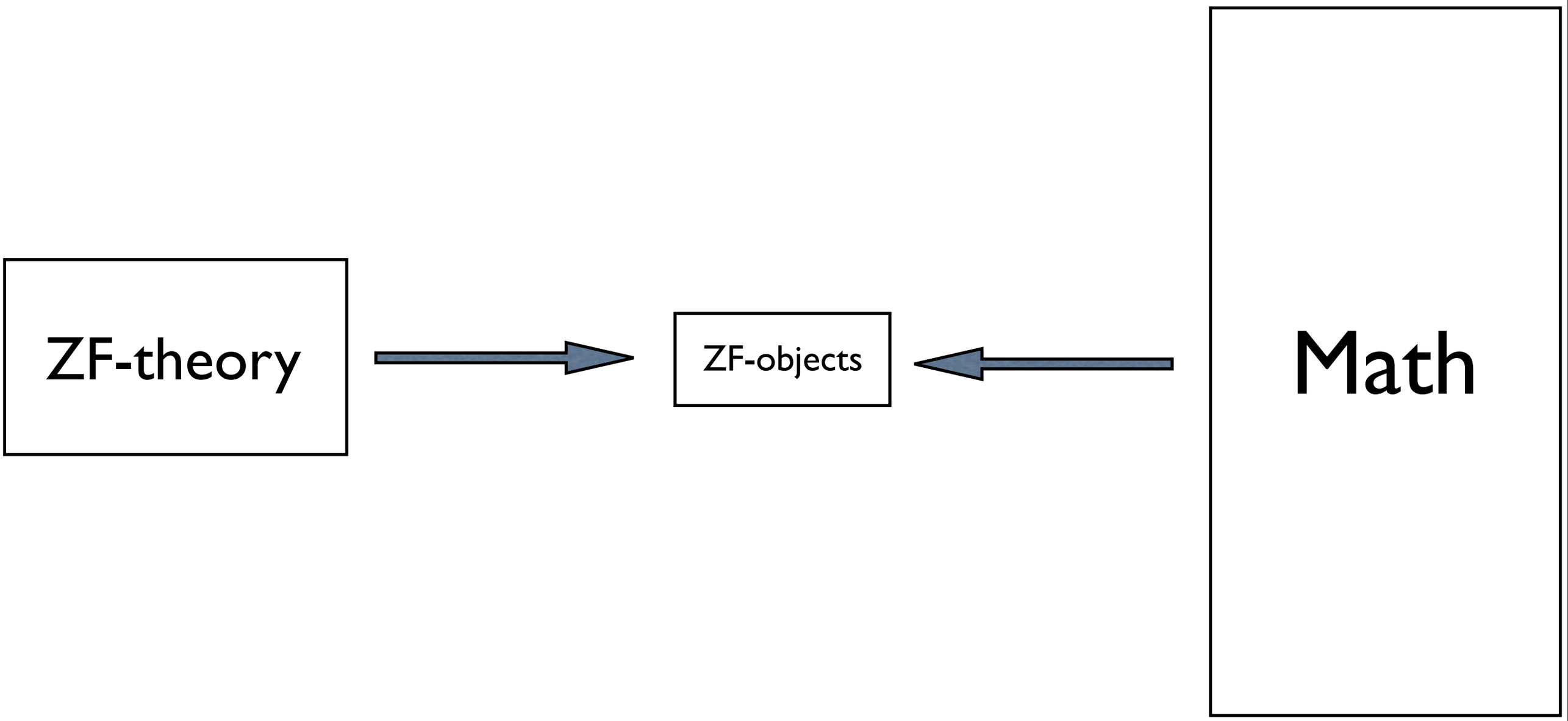
1. It is not very difficult to construct proof editing and proof verification software for any given formal deduction system

2. *We lack a good approach to formalization of mathematics*

What is a formalization of math?



Zermelo-Fraenkel Theory (ZF-theory)



The meeting point of formal and informal is very narrow. Encoding involves a lot of arbitrary choices. Implemented in Mizar.

The main issue which makes all current formalizations hard to use is the

Equality Problem

When mathematics is translated into the formal language objects which we intuitively perceive as being "equivalent" get completely different encodings.

It is most serious in the ZF-theory but it is also present in all other current approaches to formalization.

Equality problem example:

$$\sum_{i=1}^{n} i = n(n+1)/2$$

$i \in \mathbf{N}$ – natural numbers

?

=

$$\sum_{i=1}^{n} i = n(n+1)/2$$

$i \in \mathbf{Z}$ – integers

$$e.g. \quad 1 + \ldots + 10 = 55$$

We know these statements are equivalent because we know that natural numbers and non-negative integers are "the same". In the ZF-theory they are not.

Let us look more carefully at what we mean by saying that two sets are “the same”.

The first non-trivial case - two sets with two elements each. Any two such sets are in some sense equivalent - if one proofs a theorem about one set with two elements it should also hold for any other.

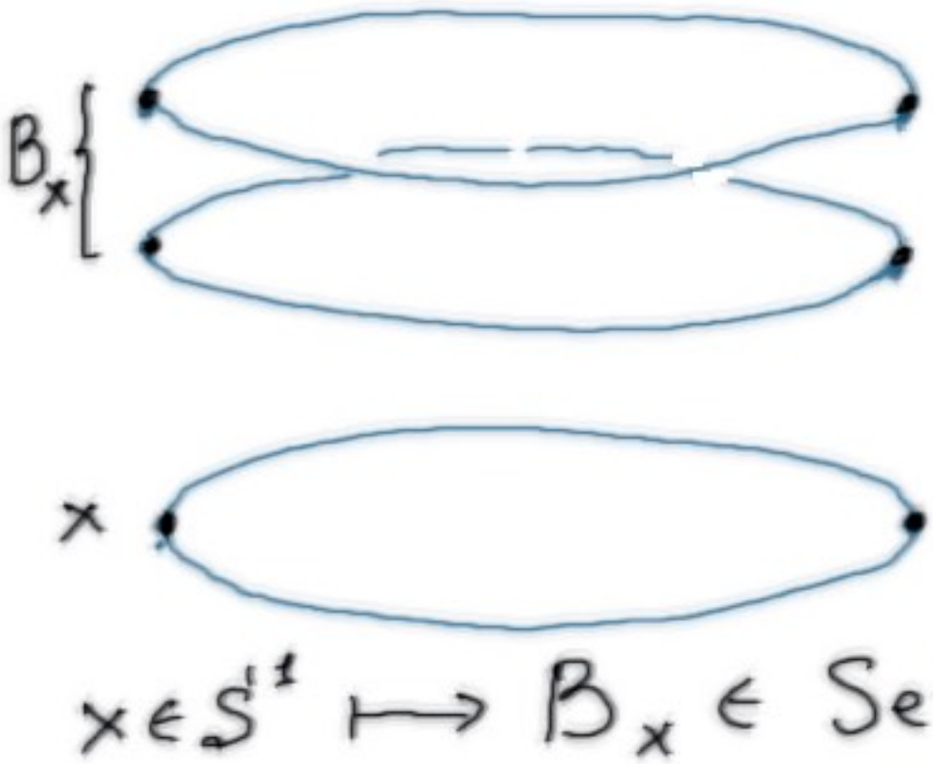
Th. The set {0,1} has 4 subsets.

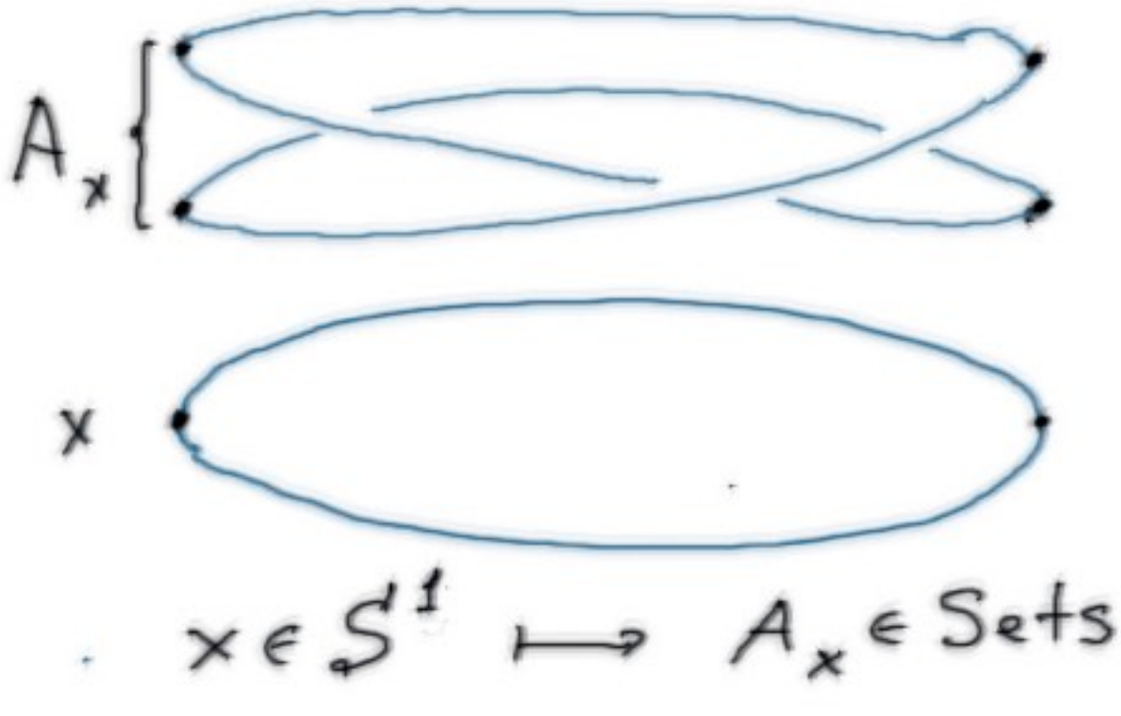
Proof: empty, {0}, {1},the whole set.

Th. The set {a,b} has 4 subsets.

Proof: --

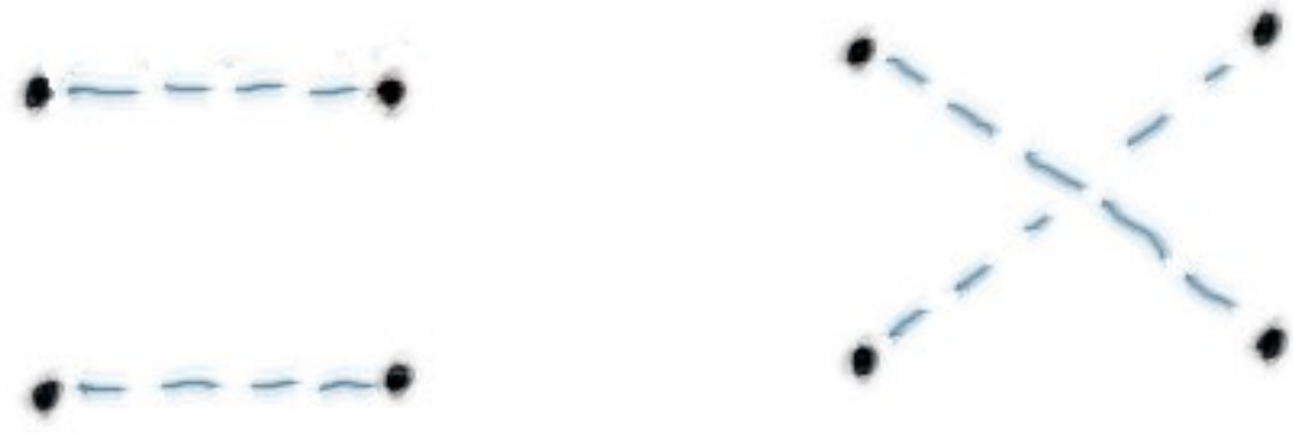
What if we declare all sets with two elements to be equal?





If all sets with two elements were equal then these two families would be equal - but they are not.

The problem is that two sets with two elements each can be identified in two different way:



We conclude that when we are talking about "composite" objects such as sets it is important to specify *how* two objects are identified.

In mathematics a particular identification between two sets or more complex objects is called an *isomorphism*. One may write

$$X \overset{\phi}{\simeq} Y$$

to say that ϕ is an isomorphism from X to Y

An isomorphism between two finite sets *exist* if and only if these sets have the same number of elements but there can be *many* different isomorphisms.

For example there are exactly two isomorphisms between any two sets with two elements. Between two sets with five elements there are 120 isomorphisms.

One of the keystones of contemporary mathematics is the

isomorphism invariance principle

For any statement P about X and any isomorphism

$$X \overset{\phi}{\simeq} X'$$

there is a statement P_ϕ about X' such that
P holds if and only if P_ϕ does.

The equality problem in formalizations comes in part from the fact that when one encodes X and X' the isomorphism is lost.

There is more to the equality problem than isomorphism invariance:

- equality is a good notion for "elements" - individuals, but fails for collections

- isomorphism is a good notions for collections of elements but fails for collections of collections.

This leads to a theory of n-equivalences which are the correct replacements of isomorphisms for such "iterated collection"

In order to avoid the equality problem a formalization should be invariant under these higher equivalences. This means that both the encoding and the semantics should respect equivalences. For the semantics arrow it means that

it should be impossible to formulate a statement which is not invariant with respect to equivalences.

Until recently no one even thought seriously about developing such an invariant formalization. Several new developments suggest that this indeed can done and in a very cool way!

First, we have an observation going back to Alexander Grothendieck:

Formalism of higher equivalences (theory of higher groupoids) = Homotopy theory (theory of shapes up to a deformation)

Combined with some other ideas it leads to an encoding of mathematics in terms of the homotopy theory. Unlike the usual encoding in terms of the set theory this one respects equivalences. This is not very unexpected yet.

To use this encoding we put homotopy theory in the middle of the formalization scheme. Then we need the left half:

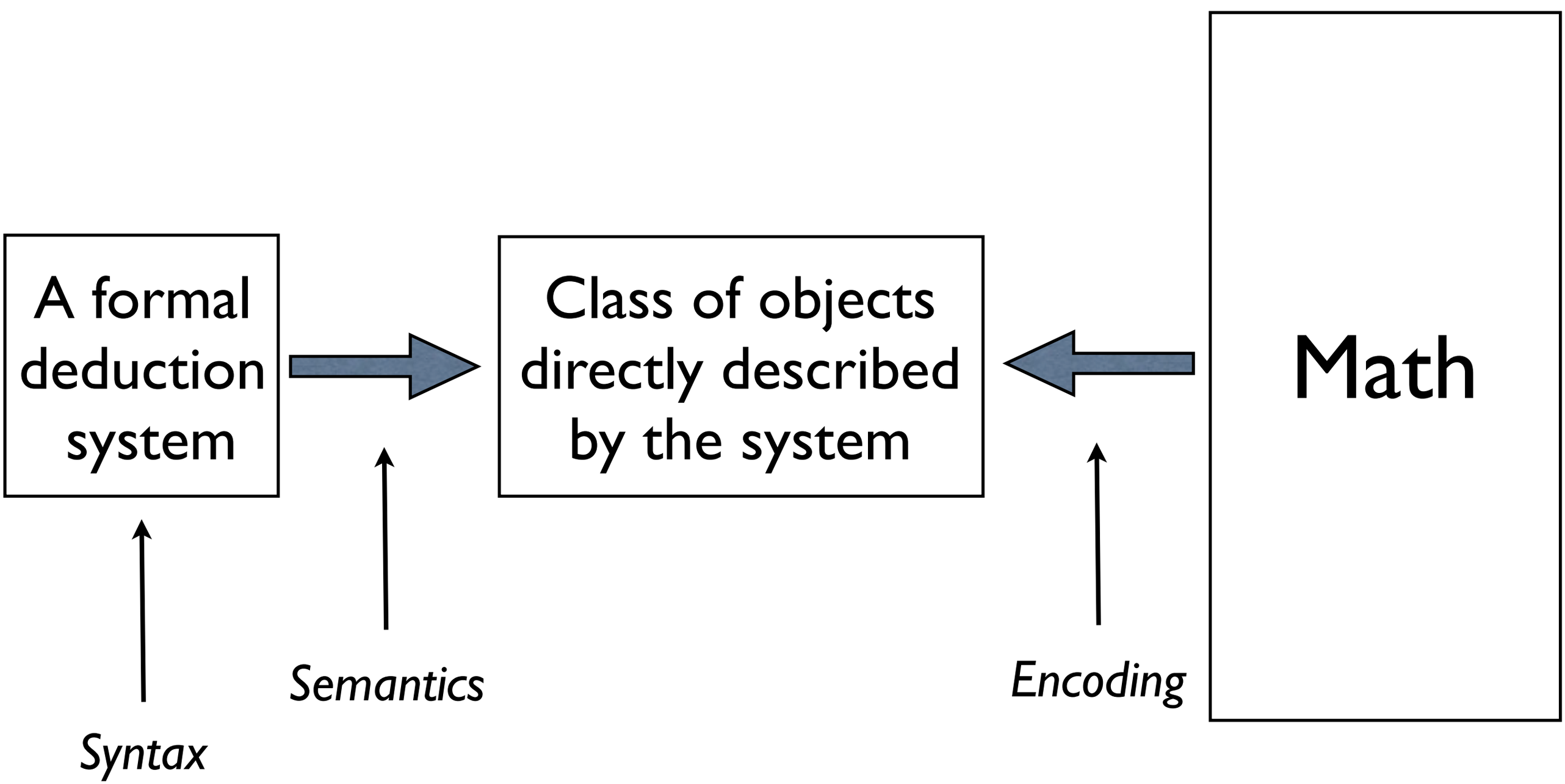
- A formal deduction system
- Semantics arrow with values in the homotopy theory

Turns out that prototypes of such deduction system already exists in the form of

Dependent type systems

- a little known subfield of theoretical computer science which goes back to the work of Nicolaas de Bruijn on the Automath

What is a formalization of math?



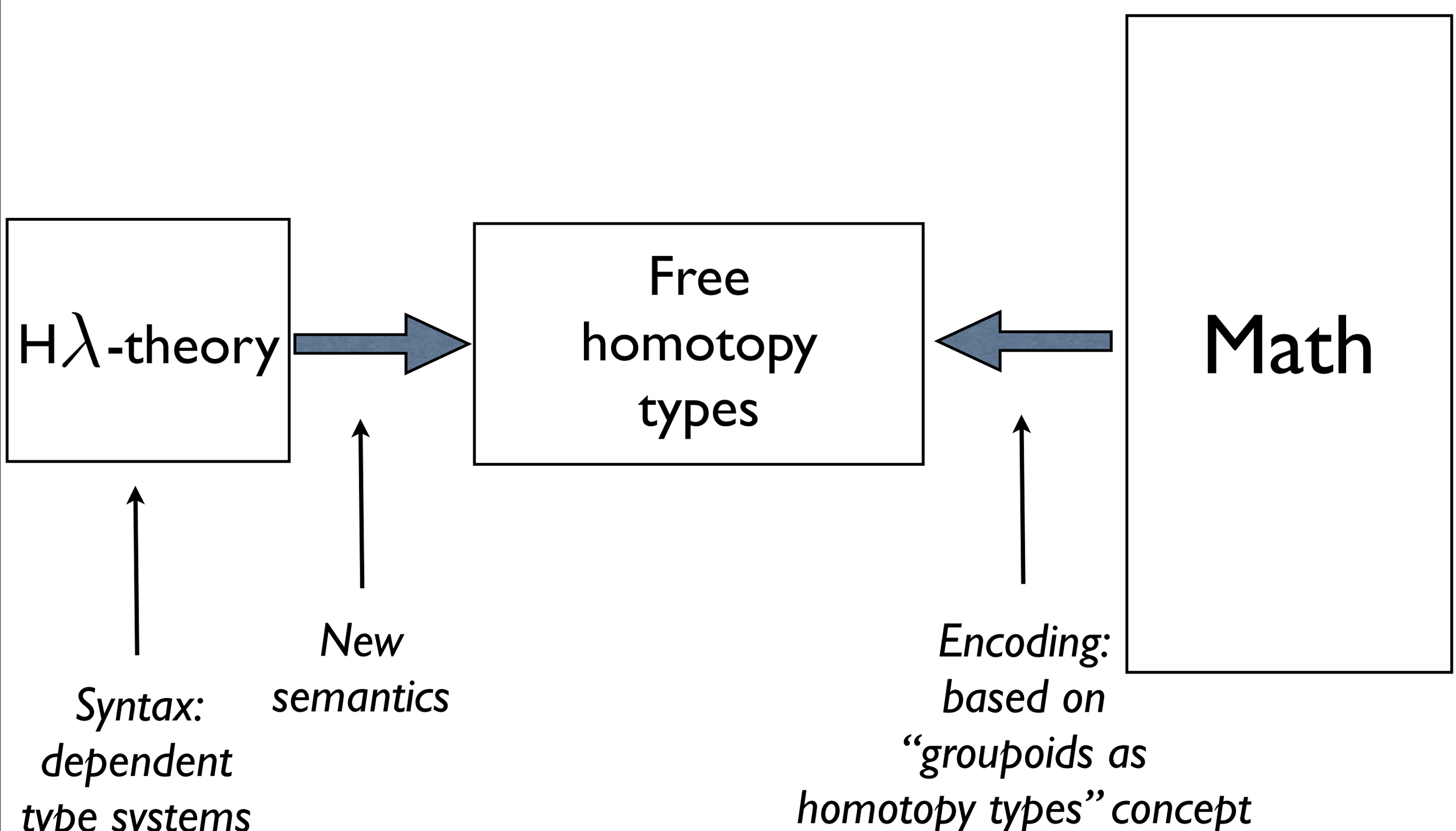
Dependent type systems are mostly used in computer science in software verification (Intel, NASA).

Never really took off in the foundations of mathematics because no one could construct a semantics.

Now it seems that the semantics for some dependent type systems naturally lands in the homotopy theory.

This observations bounds all the pieces together and leads to the following tentative picture:

Homotopy λ- calculus



This formalization should change the field of automated proof verification in pure mathematics and eventually bring a solution to the problems mentioned at the beginning of the lecture.

The end.