ON THE CONSISTENCY, COMPLETENESS, AND CORRECTNESS PROBLEMS

by

Harvey Friedman*
Ohio State University

May, 1979

Revised June, 1979

Gödel's second incompleteness theorem rules out the possibility of constructing a syntactically definite comprehensive model[1] of mathematical practice in which a consistency proof for the model can be given within the model, provided that the model is based on arbitrarily long finite derivations.

However, Gödel's second incompleteness theorem and nothing we have been able to prove rules out the possibility of constructing a syntactically definite comprehensive model of mathematical practice in which a consistency proof for the model can be given within the model, where the model is instead based on a realistic upper bound on the size of derivations. I do believe that a suitable finite incompleteness theorem can be established to rule this out, but I have only obtained partial results.

Specifically, here is an outline of a possible program which would more or less establish the consistency of mathematics within mathematics.

Firstly, a syntactically precise model of mathematics is constructed based on the axioms of ZFC such that the number of symbols in a formal derivation

---

*This research was partially supported by NSF grant MCS 78-02558.

[1]Model in the general scientific sense, not in the sense of a relational structure.

of a theorem is closely tied to the number of symbols an author would use in a complete proof of that theorem. The usual axioms and rules of inference are inadequate in this regard, particularly since they do not allow the introduction of new constant, relation or function symbols given by explicit definition, used for the purpose of abbreviation, in the course of a derivation. This must be allowed. The logic should be given in terms of a suitable linearized natural deduction system which is generous in its single step inference rules.

Secondly, a diverse collection of mathematical proofs are formalized in this " realistic ZFC" and the number of characters is counted. Or without giving complete formalizations, the number of characters involved is estimated. For definiteness, a specific list of 200 characters is allowed for realistic ZFC.

At this point, a difficulty enters. What do we mean by, say, the Riemann mapping theorem in realistic ZFC? In this framework, the Riemann mapping theorem will be given by a series of explicit definitions (and perhaps statements of theorems justifying the well definedness of these definitions (e.g., the univalence of introduced function symbols)), ending in one particular formula being labelled the Riemann mapping theorem. A proof of the Riemann mapping theorem is to take place as a proof of its given statement from the axioms of ZFC where the explicit definitions preceding the statement of the Riemann mapping theorem are also taken as axioms. Of course, as in all proofs in realistic ZFC, further symbols will be introduced by explicit definition in the course of this proof for the purpose of abbreviation.

We give some guidelines for success in this second part of the program. A printed page of mathematics has about 1000 characters. A short book therefore

has about 100,000 characters, and a treatise has about 1,000,000 characters.

I would venture a guess that if theorems such as the Riemann mapping theorem, Jordan curve theorem, and Lebesgue's differentiation theorem come out to not more than 200,000 characters, and simpler theorems such as the fundamental theorem of algebra (elementary proof), fundamental theorem of calculus, and the fundamental theorem of arithmetic come out to no more than 50,000 characters from scratch in realistic ZFC, then it would be generally regarded as successful. But I don't really know, since we do not have any measurable experience with the actual formalization of nontrivial theorems in suitable formal systems. There simply are no suitably realistic formal systems.

Thirdly, the precise syntax of realistic ZFC is formally given within realistic ZFC. Thus the symbols are defined. The terms are defined. The formulas are defined. Needed syntactic operations on the formulas are defined. The proofs are defined. In the process, new symbols are introduced by explicit definition, not only to denote the syntactic categories and operations, but also as auxiliary definitions to facilitate the development.

Assuming the success of the second part of this program, and depending on how complicated a form of realistic ZFC is used, this development should take no more than, say, 100,000 characters. Perferably 20,000 characters.

The development ends with the presentation of the desired consistency statement, asserting that realistic ZFC has no contradiction in $\leq n$ steps, where n is a specific number given in the (realistic ZFC formalization of the) decimal notation.

I think that any choice of n between, say, 10^8 and 10^{12} is relevant for the consistency problem for mathematical practice. Treatises may build, one on top of another, to perhaps a chain of 100. Of course, there may be over

time, a production of 1,000,000 treatises from 1,000,000 productive mathematicians. From the point of view of chains of treatises, we are interested in proofs in realistic ZFC with $\leq 10^8$ characters. From the point of view of total production of treatises, we are interested in proofs in realistic ZFC with $\leq 10^{12}$ characters.

Fourthly and finally, a proof totally within realistic ZFC is given of the consistency statement in part 3. If 10^8 is used in part 3, then the proof the consistency statement is to use no more than 10^8 characters and preferably much less.

An interesting possibility is that a proof of 10^8 - consistency is carried out with $\leq 10^6$ characters. Thus a proof that no chain of 100 treatises can contain a contradiction can itself be proved in one treatise. Another interesting possibility is that a proof of 10^6-consistency is carried out with $\leq 10^5$ characters. Thus a proof that no treatise can contain a contradiction can itself be proved in a short book.

Truly wild additional possibilities exist. For instance, a proof of 10^8-consistency is carried out with $\leq 10^6$ characters, <u>where the consistency proof does not use the axiom of infinity</u>!

We now present an argument which does rule out some of the most extreme possibilities.

In the course of the part 3 development, we may assume that $<$, ℓm, Prf, Neg, and Conj have been introduced, with the intention that $x < y$ iff x and y are natural numbers with x less than y. $\ell m(x)$ is the number of characters in the formula x (of realistic ZFC) if x is such a formula, $\emptyset$ o.w. $Prf(x,y,z)$ iff x is a proof of formula y in realistic ZFC with assumptions z. $Neg(x)$ is the negation of the formula x if x is

a formula; ↑ o.w. Conj(x,y) is the conjunction of the formulas x and y if they are formulas; undefined o.w.

We assume that the consistency statement formulated in part 3 is: $\sim(\exists x)(\exists y)(\mathrm{Prf}(x,\mathrm{Conj}(y,\mathrm{Neg}(y)),\emptyset)\ \&\ \ell n(x) \leq t)$, where t is a closed term involving simple arithmetical language introduced in the development in part 3.

To each formula φ of **realistic** ZFC, there will be a closed term $\#(\varphi)$ using only symbols in the part 3 development such that $\#(\varphi)$ will represent φ in a sense made clear below. The number of characters in $\#(\varphi)$ will be at most, say, four times the number of characters in φ. The process of passing from a formula φ to the closed term $\#(\varphi)$ can be easily formalized in **realistic** ZFC, as well as the process of syntactic substitution. We augment the part 3 development by developing the formal definition of the function symbol Sub. The intention is that Sub(x) is the formula resulting from replacing every free occurrence of the variable 'y' in the formula x by the closed term $\#(x)$. This augmented part 3 development is an insignificant extension, and so we retain our upper estimate of 100,000 characters.

We also have a closed term $\#(\vec{\varphi})$ for representing any possibly empty finite sequence of formulas $\vec{\varphi}$ in **realistic** ZFC. The intention is that $\#(\wedge) = \emptyset$, and that $\#(\vec{\varphi})$ is the sequence of $\#(\varphi_i)$, where $\vec{\varphi} = (\varphi_1,\ldots,\varphi_n)$. We let Dev be the augmented part 3 development, viewed as the sequence of its definitions, and we assume that this extension of $\#$ to sequences is part of this augmented part 3 development.

Consider the formula $\varphi = \sim(\exists x)(\mathrm{Prf}(x,\mathrm{Sub}(y),\#(\mathrm{Dev}))\ \&\ \ell n(x) \leq t)$. Next consider the sentence $\psi = \varphi^{y}_{\#(\varphi)} = \sim(\exists x)(\mathrm{Prf}(x,\mathrm{Sub}(\#(\varphi)),\#(\mathrm{Dev}))\ \&\ \ell n(x) \leq t)$. The length of this sentence is roughly 200,000 characters.

Now the number t is to denote an actual natural number $|t|$.

Assume by way of contradiction that ψ is provable in realistic ZFC with Dev, using $\leq |t|$ characters. Since $\#(\psi)$ is to represent ψ in realistic ZFC, we are able to prove $(\exists x)(\mathrm{Prf}(x,\#(\psi),\#(\mathrm{Dev})))$ in realistic ZFC with Dev. The delicate point is what upper bound on an x can be obtained in realistic ZFC with Dev?

We will assume that the introduction of associative function symbols such as concatenation, addition and multiplication are allowed. With such function symbols, which are of course binary, no parentheses is required in the syntax when stringed together. This is especially useful in the formal representation of finite sequences. Thus $x^\frown y$ is the sequence of length 2 whose first term is x and whose second term is y.

Since the proof of ψ is liable to involve many introductions of new symbols defined in terms of old, we are going to have to establish that these new symbols are indeed new. Since this involves formally matching pairs to prove that the proof of ψ is a proof of ψ, we are running into perhaps t^2. My feeling is that t^2 will do, and we are able to prove $(\exists x)(\mathrm{Prf}(x,\#(\psi),\#(\mathrm{Dev})) \;\&\; \ell h(x) \leq t^2)$, which is not enough for a contradiction.

Instead assume that ψ is provable in realistic ZFC with Dev, using $\leq \sqrt{|t|}$ characters. We then can prove $(\exists x)(\mathrm{Prf}(x,\#(\psi),\#(\mathrm{Dev})) \;\&\; x \leq t)$ in realistic ZFC with Dev. Since $\#(\psi) = \mathrm{Sub}(\#(\varphi))$ is provable in realistic ZFC with Dev, we have a contradiction.

We have thus "shown" that realistic ZFC with Dev does not prove ψ with $\leq \sqrt{|t|}$ characters.

Let $\psi = \sim\sigma$, and so $\sigma = (\exists x)(\mathrm{Prf}(x,\mathrm{Sub}(\#(\varphi)),\#(\mathrm{Dev})) \;\&\; \ell h(x) < t)$. Consider $\sigma \to (\exists x)(\mathrm{Prf}(x,\#(\sigma),\#(\mathrm{Dev})) \;\&\; \ell h(x) \leq t^2)$. By the same considerations

as above, this implication is provable in **realistic** ZFC with Dev, and the number of characters is tied to the number of characters in the formula. We give the estimate of 1,000,000.

Now we obviously have a proof of $\sigma \to (\exists x)(\mathrm{Prf}(x, \mathrm{Neg}(\#(\sigma)), \#(\mathrm{Dev})) \ \&\ x \leq t)$ in **realistic** ZFC with Dev using $\leq 1{,}000{,}000$ characters since $\mathrm{Neg}(\#(\sigma)) = \mathrm{Sub}(\#(\varphi))$ can be proved with $\leq 1{,}000{,}000$ characters. Hence we have a proof of $\sigma \to (\exists x)(\exists y)(\mathrm{Prf}(x, \mathrm{Conj}(y, \mathrm{Neg}(y)), \emptyset) \ \&\ \ell h(x) \leq t^2 + t + 1)$ in **realistic** ZFC with Dev using $\leq 3{,}000{,}000$ characters.

Recall that σ cannot be refuted using $\leq \sqrt{|t|}$ characters. Hence $\sim(\exists x)(\exists y)(\mathrm{Prf}(x, \mathrm{Conj}(y, \mathrm{Neg}(y)), \emptyset) \ \&\ \ell h(x) \leq t^2 + t + 1)$ cannot be proved in **realistic** ZFC with Dev using $\leq \sqrt{|t|} - 3{,}000{,}000$ characters.

Our "result" begins to have significance for the consistency statement around 10^{30} characters. Thus with some lack of confidence, we give the following proposition.

THESIS. If the consistency statement in part 3 of the program is for proofs with $n \geq 10^{30}$ characters, then the consistency proof must use at least $n^{1/4}$ characters.

I hope that someone is willing to do the necessary hard thinking for a really satisfactory discussion. Nevertheless, it is obvious that we have come nowhere near to ruling out the interesting possibilities raised earlier.

The beginnings of the construction of truly "realistic" systems can be found in Syntax and semantics of mathematical text, April 1977, unpublished.

This method of proof of course yields finite forms of Gödels second incompleteness theorem for the usual unrealistic formal systems.

We also conjecture that for any reasonable system, a proof of consistency for proofs with $\leq n$ characters requires roughly 2^n characters. More specifically, for every k there is an m such that for all $n \geq m$, any proof of n-consistency requires more than n^k characters. Observe that this conjecture implies $P \neq NP$.

We now wish to consider the completeness and correctness problems.

A good way of discussing the complexity of sentences is in terms of Turing machines. Thus we now assume that the development of Turing machines is part of Dev. Or also assume that the development of your favorite reasonable programming language is part of Dev.

Consider the following properties of n, m, p, q, t, and systems T, S which include Dev.

1. (Consistency). There is a proof in T using $\leq n$ characters that "there is no proof in $\overline{S}$ of the conjunction of a sentence with its negation using $\leq \overline{m}$ characters."

2. ($\exists$-completeness). For every program P using $\leq p$ characters which eventually halts at the empty input, it is provable in T using $\leq n$ characters that "$\overline{P}$ eventually halts at the empty input."

3. ($\forall$-completeness). For every program P using $\leq p$ characters which never halts at the empty input, it is provable in T using $\leq n$ characters that "$\overline{P}$ never halts at the empty input."

4. (Bounded completeness). For every program P using $\leq p$ characters which halts at the empty input in $\leq t$ steps, it is provable in T using $\leq n$ characters that "$\overline{P}$ halts at the empty input in $\leq \overline{t}$ steps."

5. (Bounded correctness). There is a proof in T using $\leq n$ characters that "for every program P using $\leq \bar{p}$ characters, if there is a proof in $\bar{S}$ using $\leq \bar{m}$ characters that" $\bar{P}$ halts at the empty input in $\leq \bar{\bar{t}}$ steps," then P halts at the empty input in $\leq \bar{t}$ steps."

6. (Uniform bounded correctness). There is a proof in T using $\leq n$ characters that "for every program P using $\leq \bar{p}$ characters and inputs x using $\leq \bar{q}$ characters, if there is a proof in $\bar{S}$ using $\leq \bar{m}$ characters that "$\bar{P}$ halts at the input $\bar{x}$ in $\leq \bar{\bar{t}}$ steps," then P halts at the input x in $\leq \bar{t}$ steps."

7. ($\exists$-correctness). There is a proof in T using $\leq n$ characters that "for every program P using $\leq \bar{p}$ characters, if there is a proof in $\bar{S}$ using $\leq \bar{m}$ characters that "$\bar{P}$ eventually halts at the empty input," then P eventually halts at the empty input."

8. (Uniform $\exists$-correctness). There is a proof in T using $\leq n$ characters that "for every program P using $\leq \bar{p}$ characters and inputs x using $\leq \bar{q}$ characters, if there is a proof in $\bar{S}$ using $\leq \bar{m}$ characters that "$\bar{P}$ eventually halts at the input $\bar{x}$," then P eventually halts at the input x."

9. ($\forall$-correctness). There is a proof in T using $\leq n$ characters that "for every program P using $\leq \bar{p}$ characters, if there is a proof in $\bar{S}$ using $\leq \bar{m}$ characters that "$\bar{P}$ never halts at the empty input," then P never halts at the empty input."

10. (Uniform $\forall$-correctness). There is a proof in T using $\leq n$ characters that "for every program P using $\leq \bar{p}$ characters and inputs x using $\leq \bar{q}$ characters, if there is a proof in $\bar{S}$ using $\leq \bar{m}$ characters that "$\bar{P}$ never

halts at the input $\overline{x}$," then P never halts at the input x."

11. ($\forall\exists$-correctness). There is a proof in T using $\leq n$ characters that "for every program P using $\leq \overline{p}$ characters, if there is a proof in $\overline{S}$ using $\leq \overline{m}$ characters that "$\overline{P}$ eventually halts at every input," then P eventually halts at every input."

12. (Uniform $\forall\exists$-correctness). There is a proof in T using $\leq n$ characters that "for every program P using $\leq \overline{p}$ characters and first input x using $\leq \overline{q}$ characters, if there is a proof in $\overline{S}$ using $\leq \overline{m}$ characters that "$\overline{P}$ eventually halts using $\overline{x}$ as the first input and any second input," then P eventually halts using x as the first input and any second input."

13. ($\exists\forall$-correctness). There is a proof in T using $\leq n$ characters that "for every program P using $\leq \overline{p}$ characters, if there is a proof in $\overline{S}$ using $\leq \overline{m}$ characters that "$\overline{P}$ never halts at some input," then P never halts at some input."

14. (Uniform $\exists\forall$-correctness). There is a proof in T using $\leq n$ characters that "for every program P using $\leq \overline{p}$ characters and first input x using $\leq \overline{q}$ characters, if there is a proof in $\overline{S}$ using $\leq \overline{m}$ characters that "$\overline{P}$ never halts using $\overline{x}$ as the first input and some second input," then P never halts using x as the first input and some second input."

An interesting choice of numbers for these might be: $n = 10^6$, $m = 10^8$, $p = 10^5$, $q = 10^7$, $t = 10^{10}$, and $S = T = ZFC + Dev$. It is easy to refute 2, 3, 7-10 under this choice. We know nothing else of any interest, except that from what we sketched above, n must be at least roughly $m^{1/4}$ (for $n \geq 10^{30}$). And again, it does not help matters if S is instead T without infinity.