FRANCISCO ANTONIO DORIA

IS THERE A SIMPLE, PEDESTRIAN ARITHMETIC SENTENCE WHICH IS INDEPENDENT OF ZFC?

ABSTRACT. We show that the $P < NP$ conjecture can be formulated as a Π_2^0 sentence, and explore some of the consequences of that fact. This paper summarizes recent work by the author with N. C. A. da Costa on the $P < NP$ conjecture and on the possibility that this sentence is independent of ZFC supposed consistent.

1. INTRODUCTION

Can we find a simple, pedestrian, fully arithmetic question that can be shown to be independent of the axioms of set theory? A question so simple that can be understood by the nonmathematician, and yet whose solution leads us to deep and difficult foundational questions?

The present paper suggests one such candidate, the $P < NP$ question. A few years ago the author was asked by his department's secretary the following question, which we formulate as a short tale:

Mrs. H. is a gentle and able lady who has long been the secretary of a large university department. Every semester Mrs. H. is confronted with the following problem: there are courses to be taught, professors to be distributed among different classes of students, large and small classes, and a shortage of classrooms of varying sizes. She fixes a minimum acceptable level of overlap among classes and students and sets down (in a tentative way) to get the best possible schedule, given that minimum desired overlap. It's a tiresome task, and in most cases (when there are many new professors, or when the dean changes the classroom allocation system) Mrs. H. feels that she has to check every conceivable scheduling before she is able to reach a conclusion. In despair she asks a professor whom she knows has a degree in math: "tell me, can't you find in your math an easy way of scheduling our classes with a minimum level of overlap among them?"

The formal equivalent of Mrs. H's question is of course the $P = NP$ question (the negation of $P < NP$),[1] whose precise, formal statement we discuss below.

Mrs. H's difficulties arise because the number of cases to be examined blow up in an exponential way as the number of classes, professors and classrooms increases. However we must give precision to her query,

…can't you find an easy way of scheduling our classes with a minimum level of overlap among them?

This is formalized by supposing that the easy answer should be time-polynomial on the input. Granted this (and given a coding assumption, that every input and output is binarily coded) we can formalize Mrs. H's question as the $P = NP$ question (see below).

We conjecture that $P = NP$ and $P < NP$ are independent of the axioms of Zermelo–Fraenkel set theory, the axiom of choice included (ZFC). The present paper wishes to present our case: to give the reasons why we think so, and to sketch a program to settle the matter in the desired direction.

This conjecture seems to run against the feelings of most researchers in the area, who think that the matter will be eventually decided in the direction of ZFC proving that $P < NP$. Yet there is a result by DeMillo and Lipton (1980) (see the comments in Johnson (1990)) that proves the consistency of $P = NP$ with fragments of number theory. There is, moreover, the well-known Baker et al. (1975) relativization result These rather counterintuitive results could be reconciled if we had an independence result for $P = NP$.

1.1. *A Very Short Historical Note*

For the references see Machtey and Young (1979). Problems in the NP class can always be settled, but it's difficult to get a solution in the general case. However, if we "guess" a solution, it is easy to check that it actually settles the problem. Now substitute "difficult" for "exponential-time in the length of the input", and "easy" for "polynomial-time in the length of the input", and we (intuitively) get the NP class.

Class importance has to do with the fact that several everyday problems appear in it, such as Mrs. H's problem above. Characterizations and first examples were given by Cook (1971) and Karp (1972). A first review was given in the book by Garey and Johnson (1978). Finally, a first important result was obtained by Baker et al. (1975).

2. $P < NP$ AS A Π_2^0-SENTENCE

We can intuitively formulate the main characteristics of a problem in the NP-class:

It is easy to check whether a possible solution satisfies a given instance of the problem, but the only known ways to actually find a solution, in the general case, are equivalent to the following procedure: list all possible solutions and test each one against the instance of the problem.

That procedure blows up exponentially as a function of the size (expressed as the length) of the input.

If we understand that "polynomial time on the length" (for a binarily coded expression) adequately translates "easy" then we can formulate the $\mathcal{A}_R$ class of nondeterministic polynomial, or NP-problems, as follows: let p_R be a fixed positive-definite polynomial that depends on R; let $R(x, y)$ be a recursive, polynomial predicate such that we can explicitly compute the Gödel number of the corresponding characteristic function, and let $|x|$ be the length of x, expressed in binary notation in the canonical enumeration. Then (Baker et al., 1975; Johnson 1990):

DEFINITION 2.1. $x \in \mathcal{A}_R \leftrightarrow_{\text{Def}} \exists y(|y| \leq p_R(|x|) \wedge R(x, y))$. □

x is the instance of the problem, and y the solution. $R(x, y)$ gives the "easy" (polynomial) test of y; if it holds, then y "settles" x. The bounding condition $|y| \leq p_R(|x|)$ ensures that if we list all possible solutions for x (all possible y), that list will roughly equal $2^{p_R(|x|)}$. Therefore, given that approach, we will need an exponential time (in the length $|x|$ of the input) to settle instance x.

We can define:

DEFINITION 2.2. $NP = \{\mathcal{A}_R\colon \text{all } R\}$. □

Now there is a trick that we can use (Baker et al. 1975) in order to effectively enumerate expressions for all polynomial machines.

Then let P_m be the polynomial machine of index m according to that enumeration. If x is input to P_m, we can form the predicate:

DEFINITION 2.3. $A_R(m, x) \leftrightarrow_{\text{Def}} R(x, \mathsf{P}_m(x))$. □

(We can informally read $A_R(m, x)$ as "polynomial machine of index m settles instance x of $\mathcal{A}_R$.")

Finally, $P < NP$ is, roughly, *there is no easy way to settle all instances of a problem* $\mathcal{A}_R$. More precisely, no polynomial algorithm settles all instances of $\mathcal{A}_R$, or,

DEFINITION 2.4. $P < NP \leftrightarrow \forall m \exists x \neg A_R(m, x)$. □

So, $P < NP$ is given by a Π_2^0 sentence. If we put, DEFINITION 2.5. $f_R(m) = \mu_x \neg A_R(m, x)$. $\square$

We conclude,

COROLLARY 2.6. $P < NP \leftrightarrow f_R$ is total. $\square$

$\mathcal{T}$-provably Total Recursive Functions

$\mathcal{T}$ is here either Peano arithmetic (PA) or Zermelo–Fraenkel set theory with the full Axiom of Choice (ZFC). The concept we now introduce originated in Kreisel (1951, 1952): roughly a recursive function f is $\mathcal{T}$-provably recursive if, for some Gödel number e:

1. $\mathcal{T}$ proves that e is the Gödel number of f, and
2. For each x, $\mathcal{T}$ proves that the computation of $f(x)$ converges.

In what follows we suppose that all variables are restricted to ω, the set of natural numbers. Formally,

DEFINITION 2.7. A $\mathcal{T}$-unary function f is $\mathcal{T}$**-provably total unary recursive** if for some Gödel number e_f for f, $\mathcal{T} \vdash \forall x \exists z (T(e_f, x, z) \wedge \forall y (f(y) = \{e_f\}(y))$. $\square$

It follows,

COROLLARY 2.8. If f is a $\mathcal{T}$-provably total unary recursive function, then there is a g, such that g is $\mathcal{T}$-provably total unary recursive function and g bounds f, that is, for each x, $g(x) \geq f(x)$. $\square$

Therefore the operation time of f is bounded by the operation time of the bounding function, and as a result we have a proof in $\mathcal{T}$ that every computation of f converges.

However the concept of $\mathcal{T}$-provably total recursive unary functions isn't a trivial one: list all $\mathcal{T}$-provably total unary recursive functions. Then diagonalize over that list. We get a total unary recursive function which isn't in the list.

Moreover, those non-$\mathcal{T}$-provably total unary functions 'top' all $\mathcal{T}$-provably total unary functions:

- Say that f *dominates* g if $f(x) \geq g(x)$ but for a finite number of values. Then if g isn't dominated by f, g overshoots infinitely many times through f.
- Corollary 2.8 implies that if g isn't dominated by any $\mathcal{T}$-provably total recursive unary function, then it isn't a $\mathcal{T}$-provably total unary function.

That is,

COROLLARY 2.9. If, for any $\mathcal{T}$-provably total recursive unary function f we have that g overshoots through f infinitely many times (that is, for infinitely many x, $g(x) > f(x)$), then g isn't a $\mathcal{T}$-provably total recursive unary function. □

3. PROVABILITY OF $P < NP$ IN A GIVEN AXIOMATIC SYSTEM $\mathcal{T}$ AND $\mathcal{T}$-PROVABLY TOTAL RECURSIVE FUNCTIONS

Why do we need that quite restrictive concept? Because of the following: roughly, PA (or for that matter ZFC) proves that $P < NP$ if and only if f_R is provably total unary recursive in this strict sense in the corresponding theory. For:

- We can explicitly compute a Gödel number for the characteristic function of R.
- Follows that we can explicitly compute a Gödel number for f_R.
- Let e be that Gödel number. Then $\{e\} = f_R$.
- As the proof of $P < NP$ is the proof of the formal sentence $\forall x \exists y [f_R(x) = y]$, we get our conclusion.

Therefore we can find the status of the Π^0_2 sentence that formalizes our intuition about $P < NP$ if and only if we can settle whether f_R is $\mathcal{T}$-provably total or not.

3.1. *Two Possible Ways to Proceed*

How can we establish the status of $P < NP$ in $\mathcal{T}$, given the preceding remarks? There are two ways to proceed:

1. We can recursively enumerate all $\mathcal{T}$-provably total recursive unary functions in $\mathcal{T}$.

   Then is there a way we can compare f_R with each one of the provably total functions in the enumeration, and decide whether f_R belongs to the list or not? (See da Costa and Doria (1999b) on that possibility.)
2. Or we can try to use Corollary 2.9 and see whether f_R is dominated by one of the $\mathcal{T}$-provably total recursive unary functions or not.

   Recall that recently Wainer (1999) extended the hierarchy of fast-growing PA-total recursive functions well into ZFC.

3.2. *Preliminary and Partial Results*

Recently Cucker, in a private exchange of e-mail messages with da Costa and Doria, constructed the following function h:

EXAMPLE 3.1. Suppose given Baker et al. (1975) a fixed, recursive enumeration

$$\mathsf{P}_0, \mathsf{P}_1, \mathsf{P}_2, \dots, \mathsf{P}_m, \dots,$$

of the polynomial Turing machines; suppose also given the canonical enumeration of binary words $\emptyset$, 0, 1, 00, 01, 10, 11, 000, 001,... which code the empty word itself $\emptyset$ and the integers 0, 1, 2,....

Let $\mathsf{P}_m(x)$ be one such polynomial Turing machine; it inputs binarily coded numbers x. Let 2^x be the exponential function, computed by the usual exponential machine. Form the predicate

$$C(m, x) \leftrightarrow_{\text{Def}} \mathsf{P}_m(x) < 2^x.$$

Define the function $h(m) = \mu_x C(m, x)$. Of course h is intuitively total recursive.

Is it PA-provably total recursive? ZFC-provably total? □

Recent work (da Costa and Doria (1999a) suggests that this isn't the case. Now there is a recursive, partial map $h \to f_R$ in such a way that we can show that the fast-growing "peaks" of the function h appear in a similar way within f_R. However h is intuitively total (total in the standard model for arithmetic) while there is no clear indication that f_R is intuitively total.

REMARK 3.2. Let's elaborate on that:

- Suppose that 2^x represents the exponential Turing machine that computes the same function for argument x.
- Then the Turing machine P^a that:

  a) Inputs x and outputs 2^x, for $x \leq a$, a a constant;
  b) Inputs x and outputs 0, for all $x > a$,

  is polynomial.
- If F is any total recursive function, then the infinite family of Turing machines $\mathsf{P}^{\mathsf{F}(n)}$, $n = 0, 1, 2, \dots$, is a family of polynomial machines.
- Now let T be an exponential machine that settles all instances of $\mathcal{A}_R$. Then the machine Q^a, given by:

  a) Inputs x and outputs $\mathsf{T}(x)$, for $x \leq a$, a a constant;
  b) Inputs x and outputs 0, for all $x > a$,

is again polynomial. (This machine Q^a settles all instances of $\mathcal{A}_R$ up to $x = a$.)

- And again, for F as above, the family of Turing machines $\mathsf{Q}^{\mathsf{F}(n)}$, $n = 0, 1, 2, \ldots$, is a family of polynomial machines. They settle the instances in $\mathcal{A}_R$ up to the bound $\mathsf{F}(n)$, each n.

Each $\mathsf{P}^{\mathsf{F}(n)}$ and $\mathsf{Q}^{\mathsf{F}(n)}$ can be given an index in the enumeration referred to in the beginning of Example 3.1. If every two consecutive elements of the sequence can be kept "reasonably close" in the above indexing for polynomial machines, no matter how fast-growing is F, then we have functions h and f_R that may overshoot infinitely many times above any prescribed function in the Wainer (1999) hierarchy.

For details see da Costa and Doria (1999a,b). □

ACKNOWLEDGMENTS

This work summarizes ideas from a research project developed with N. C. A. da Costa. The author wishes to thank Fapesp for support as a Visiting Professor at the Institute for Advanced Studies, University of São Paulo. He also acknowledges scholarships and grants from CNPq (Brazil) and from UFRJ, from where he is on leave.

We are indebted to S. S. Wainer and to F. Cucker for several e-mail exchanges; and of course to our dear friend Newton da Costa for the adventurous, thorny, exploration of these fascinating mathematical landscapes.

ACKNOWLEDGEMENTS

Partially supported by FAPESP and CNPq. Permanent address for F. A. Doria: Research Center for Mathematical Theories of Communication, School of Communications, Federal University at Rio de Janeiro, Av. Pasteur, 250. 22295-900 Rio RJ Brazil.

NOTES

[1] For a preliminary discussion of the ideas of da Costa and Doria on these matters see Doria (1996); for the equivalence between an allocation problem such as Mrs. H's and the satisfiability problem, for instance, see Machtey and Young (1979).

REFERENCES

Baker, T., J. Gill, and R. Solovay: 1975, 'Relativizations of the $P = ?NP$ question', *SIAM Journal of Computing* **4**, 431.

da Costa, N. C. A. and F. A. Doria: 1999a, 'On Total Recursive, But Not $\mathcal{T}$-Total Recursive Functions', preprint IEA-USP.

da Costa, N. C. A. and F. A. Doria: 1999b, "Fast-Growing Functions Defined over Bounded Turing Machines in ZFC: A Consistency Result for Strictly Partial Functions', preprint IEA-USP.

de Millo, R. A. and R. J. Lipton: 1980, 'The Consistency of "$P = NP$" and Related Problems with Fragments of Number Theory', *Proc. 12th ACM Symp. on the Theory of Computing*, 45.

Doria, F. A.: 1996, *Logique et Analyse* **153/154**, 165.

Johnson, D. S.: 1990, 'A Catalogue of Complexity Classes', in J. van Leeuwen (ed.), *Handbook of Theoretical Computer Science*, Elsevier.

Ketonen, J. and R. Solovay: 1981, 'Rapidly Rising Ramsey Functions', *Annals of Mathematics* **113**, 267.

Kreisel, G.: 1951, 'On the Interpretation of Non-Finitist Proofs, I and II', *J. Symbol. Logic* **16**, 241; **17**, 43 (1952).

Machtey, M. and P. Young: 1979, *An Introduction to the General Theory of Algorithms*, North-Holland.

Wainer, S. S.: 1970, *Arch. Math. Logik* **13**, 136.

Wainer, S. S.: 1999, 'Accessible Recursive Functions', preprint, Department of Mathematics, University of Leeds.

Research Group on Logic and Foundations
Institute for Advanced Studies
University of São Paulo
Brazil
E-mail: fadoria@rio.com.br