1. Introduction

The distinction between existence and knowledge, when applied to the phenomenon of undecidability in theory of computation [1, fn 5] has a role antithetical, in our views, to the same distinction in criticisms to ideal mathematics. For let us consider the following kind of inference, that we call a proof by undecidable cases:

it has been proved finitarily that
(i) if $A \models \alpha$, then $B$; and that (ii) if $A \not\models \alpha$, then $B$; hence $B$,

where $A$ is a formula from a r.e. language and $\alpha$ is a non r.e. class. It is clear that if $A$ is f.i. $\{e\}(x)$↑, $B$ is not the conclusion of a potential infinity of Gedankenexperimente; as one cannot say, for each $e$ and for each $x$, whether (i) or (ii) has to be checked. Thus such a proof lies outside finitary traditional frontiers.

An argument could however be imagined against this evidence. Let us remind submarily the old motivation for consistency proofs. Reference to Kant [2] was explicit. The infinite is a (mathematical) idea, a concept of the reason, which can be justified only in terms of logical possibility (Möglichkeit), that is to say consistency. However (and this is perhaps the philosophic new by Hilbert) this possibility can be proved only at a scientific (Kant's real, wirklich) level, adding to logic certain intuitive representations (Vorstellungen), given by finitism. Now, we feel that, in a sense, metamathematical manipulations have the same degree of immediateness than Turing machines and the fact that either a given machine stops for a given input, or it does not. Thus, we incline to regard results obtained with proofs by undecidable cases as (almost) finitary.

Cut elimination and hence consistency, for a theory having recursive functions theory as its intended model, can be established by means which are strictly finitary, except for the use of such proofs [3]. By the same method we show here consistency (Section 6) of a family of theories (Section 5) in which are definable (Section 7) the hyperarithmetical sets, all represented with notations (Section 3) very close to those by Kleene and by Rogers [4].
2. Preliminaries

As a rule we try to have lower case letters for non formal entities and the same letter, but capital, for their formal correspondents. So, \( x, y, \ldots \) are informal integer variables, while \( X, Y, \ldots \) are formal variables of theories to be defined. \( N, M, \ldots \) are (formal) numerals. \( H, K, \ldots \) are formal variables or numerals.

\( \bar{x} \) and \( \varphi(x) \) are the numerals for (the values of) \( x \) and \( \varphi(x) \). \( |x, y| \) is \( 2^x \cdot 3^y \). If \( z \) is \( |x, y| \), then \( z' \) is \( x \) and \( z'' \) is \( y \); else they are both \( z \).

\( \neg, \lor, \&, \rightarrow, \forall, \exists \) are informal operators (conventional meaning conveying extensions of natural language). \( - \) and \( + \) are formal symbols (to be interpreted as negation and disjunction). Formal universal quantification is shown by parentheses the variable.

As far as possible Roger's notations [4] are used or adapted according to details specified in this section.

A characterization \( c \) of recursive functions can be extended into a characterization \( rc \) of relativized functions, by adding an if-then construct, decided by a \( \varepsilon \)-test. More precisely, we can state that the programs \( (p, q, \ldots) \) of \( rc \) are given by the following rules:

(i) the programs of \( c \) are programs of \( rc \).
(ii) if \( p \) and \( q \) are programs of \( rc \), so is \( \Delta(p, q) \).

\( p(x, A) \) is both the application of \( p \) to \( x \) and to the set \( A \), and its result. The (result of the) application of \( p \) to (the result, if any, of) \( q(x, A) \) is \( pq(x, A) \).

\( \Delta(p, q)(x, A) \) should mean: if \( x \in A \) then \( pq(x, A) \) else \( p(x, A) \). Let \( rc \) be gödelized without use of positive powers of 2 and of 3. ' \( x \)' may be, with systematic ambiguity, a program of \( rc \) or the functor formalizing it ('0' is defined in Section 4). Clearly, it is decidable whether ' \( x \)' was already in \( c \) or it has been obtained by some application of rule (ii) above; accordingly, we say that \( x \) is in \( c \), or that \( x \) is in \( rc \).

If \( x \) is in \( c \), \( \phi_x \) is \( \lambda y(\lambda x'(y, A)). \psi_x^4 \) is \( \lambda y(\lambda x'(y, A)) \).

\( W(x) \) is \( \text{dom} \phi_x \), \( W(x, A) \) is \( \text{dom} \psi_x^4 \).

We are going to restrict the class of sets which can be the second argument of a relativized recursive function to the hyperarithmetical sets (hs). Accordingly, we will substitute, in the notations above, \( A \) with an integer associated to \( A \) in the system of notations of next section.

3. Notations for Hyperarithmetical Sets

We give now a system of notations for hs’s. We define at once: (i) a set \( HS \) of integers, (ii) a many-one mapping of \( HS \) on the hs’s [\( h(x) \) is the hs associated with \( x \)], and (iii) a partial ordering \( \preceq x \) of \( HS \).

The idea is to take as zeroes the r.e. sets, and as successors of \( h(x) \) all sets \( W(y, x) \). And to take as limit of an increasing sequence (of indices of) hs’s (an index of) their union.