ALONZO CHURCH: His Life, His Work and Some of His Miracles.

María Manzano. mara@gugu.usal.es
Dpto Lógica. Universidad de Salamanca.
37007 SALAMANCA, Spain.

December 1996.

Abstract

This paper is dedicated to Alonzo Church, who died in August 1995 after a long life devoted to logic. To Church we owe lambda calculus, the thesis bearing his name and the solution to the *Entscheidungsproblem*. His well-known book, *Introduction to Mathematical Logic*, vol. I, defined the subject matter of mathematical logic, the approach to be taken and the basic topics addressed. Church was also the creator of the *Journal of Symbolic Logic*, the best-known journal of the area, which he edited for several decades.

This paper is divided in three sections. The first is written in journalistic style: the story of the life of Alonzo Church is told, including some of the many anecdotes I have collected from different sources. The second part is devoted to his work, but is far from being exhaustive. The last part is more original; in it I attempt to show that Church's great discovery was lambda calculus and that his remaining contributions were mainly inspired afterthoughts in the sense that most of his contributions as well as some of his pupils derive from that initial achievement. Included are Kleene's Recursion Theory and the completeness proof of Henkin. I have added an appendix in which is presented the typed lambda calculus and a proof of the undecidability of first-order logic.

Contents

1. Prologue.

The way information flows has radically changed in recent years, as witnessed by the mere fact that we are able to use electronic mail even if we are not computer experts. If, as we used to hear, information is power, it is also true that power has been decentralized in the last few decades. (Most probably not all the power, just the least important). I was thinking about all this when I was receiving the information I needed to write my *Alonzo Church: in memoriam* through different fast channels . (To be truthful and to add a counterargument to this praise of the **information highway,** I should say that my sources of information were Leon Henkin and John Addison in the U.S. and Wilfrid Hodges in the United Kingdom, whom I respect intellectually as they deserve, and with whom I have had the chance to maintain a friendly relationship since I was their student.) What I mean with all this is: Could I have accessed this information anonymously?

And how about Church? How did he contemplate these changes, as one who not only bore witness to almost a century of logic, but also as one who had to think of himself as being directly responsible for its creation and part of its development?

I was trying to place myself in Alonzo Church's skin, trying to imagine the vertigo of such a leap, and at the same time thinking that he had not only to jump himself, but also to witness and help others doing so. Alonzo Church was one of the most distinguished creators (I would say he is the second in importance, after only Gödel) of the two fields underpinning the theoretical basis of computer science: mathematical logic and the theory of computation. Some would add that his responsibility goes beyond the merely theoretical parts: it is today widely acknowledged [1] that John von Neumann was able to design the first electronic digital computer because of the influence of the work of Church's most famous student: Alan Turing.

Curiously enough, the answer to this rhetorical question reached me merely by chance after being formulated in public[2]

I spent the spring of 1996 at the CSLI in Stanford. One day, I was talking to John Perry about Alonzo Church and he told me that John Etchemendy could tell me an interesting anecdote. This happened one day during the academic year

[1] See, Martin Davis [20].

[2] In January 1996, at the Congress **Truth, Logic, Representation and World** held in Santiago de Compostela (Spain). There, I presented *Alonzo Church: in memoriam* in a session dedicated to him.

of 1983-84. They had invited Alonzo Church to give a talk at the CSLI. At the time, CSLI had a large number of Xerox Dandelion computers, which were LISP processing machines (similar to Symbolics machines, which became popular a few years later). Etchemendy took Church aside for a few minutes to show him a Dandelion and to explain that it was indirectly based on lambda calculus. He seemed only distantly interested and when they later walked over to the building where Church was to give his talk, he told Etchemendy (no doubt to explain his evident lack of enthusiasm about the demo) that he knew nothing at all about computers, but that he had once had a student who did know quite a bit about them, that was Alan Turing.

2. Life.

Important dates and facts. Alonzo Church was born in 1903 in Washington D.C.; his father, Samuel Robbins Church was a Justice of the Municipal Court of the District of Columbia). His grandfather was Librarian of the U.S. Senate and his greatgrandfather (whose name was also Alonzo Church) was Professor of Mathematics and Astronomy at, and late President of, what is now the University of Georgia.

Alonzo Church studied at Princeton University under Oswald Veblen, a famous geometer. In 1924 Church graduated; he married Mary Kuczinski in 1925, and in 1927 he completed his Ph.D.

After this, Church was awarded a *National Research Fellowship* that he enjoyed for two years. During the first year he was at Harvard, working with Birkhoff (the algebraist) and with Huntington. At Harvard there was a notable concentration of logicians at that time: Sheffer, Lewis, and Whitehead were also members of the Faculty there.

Church spent the second year in Europe in Göttingen, working with the members of Hilbert's school; Hilbert himself and Bernays. Church also spent part of 1929 in Amsterdam to study with the intuitionist Brouwer.

When he returned from Europe in the autumn of 1929, he began to teach at Princeton. For ten years he was an assistant professor; for eight more he served as an associate professor and was finally promoted to full professor in 1947. He stayed at Princeton until 1967, when he moved to Los Angeles to avoid retirement (since in Princeton retirement was mandatory at the age of 68). He remained in California as an active teacher until 1990, when he was 87. He had a joint contract: Flint Professor of Philosophy and Professor of Mathematics. He was therefore able to serve both Departments as teacher and researcher; indeed, he kept this dual position throughout his life.

He used to spend summers in the Bahamas, where he found peace to think away from the humdrum of everyday life.

He was what today would be known as a "workaholic" throughout his life and was endowed with an insatiable intellectual appetite until his late years. In 1983, on the occasion of his 80 birthday, UCLA Monthly published an interview with him in which he said[3]:

> I will retire when one of two things happens: when I am told by the university I have to or when I'm told by my doctor I have to. I will continue as long as it can be done. It might be tomorrow, it might be a year from now, or it might be 10 years from now. But that's highly unlikely.

In this interview when asked the question: What makes him so good? Robert Yost, vice chairman of the Philosophy Department, simply replied:

> He is just smarter than anybody else.

and added: Church reads everything and forgets nothing.

At that time Church still walked to the University every day. He never wanted to have a car because he would

> spend as much time on the car as I would with my teaching and research.

He finally moved to Hudson in 1992 to live closer to his son Alonzo Church Jr. He died on 11^{th} of August 1995. The header of his obituary in the **New York Times**[4]**,** written by Nicholas Wade, was: *Alonzo Church, 92, Theorist of the limits of Mathematics*. In the London **Independent,** his obituary written by Wilfrid Hodges was accompanied by a photograph with the caption: *A truly rational mind*.

Henkin reports that Church maintained a lively interest in Logic until the end of his life: on his bedside table, there was a volume in his honor with book marks at several of the contributions, showing that he took interest in them.

[3] Stuart Wolper. [1983]. *A legend turns 80.* UCLA Monthly.

[4] This was published on 5^{th} of September 1995.

This volume was edited by Anthony Anderson and one of his students, Mikhail Zeleny. Among the contributors are Shepherdson, Shapiro, Paulson, Meyer, Kripke, Henkin and Feferman. It was meant to be a present for Church's 90 birthday, but no publisher could be found for it. (The volume on Church's table was handmade). Henkin says that because of the difficulties in finding a publisher, he himself suggested to Antony Anderson the idea of the handmade volume. In the event, this was perhaps a "fortunate" decision for Church died a few weeks later. Two of the articles of the volume were published in the **Bulletin of Symbolic Logic;** the one by Martin Davis [21] and the one by Henkin [31].

His teaching. A teaching career of 61 years is in itself extraordinary, but Church's disciples also attest that his peculiar style made his teaching something unforgettable. Indeed, he made a profound impact on all of them. As a teacher and colleague he was good-natured, patient, modest and undemonstrative, (some people report that he lacked passion). This seems to be the opinion of Alan Turing, who spent two years at Princeton as his student and also wrote his thesis under Church's supervision[5] .

His work was always very detailed, his analyses were profound and he was very, very careful. No detail was too small: everything had to be set out slowly and precisely, in logical order and in huge handwriting.

David Kaplan says that he always recommended his new graduate students to sit in on a class with Church, telling them:

> Take a class from Professor Church. It will change you. And even if you are not interested in pursuing the subjects he teaches, you will tell your grandchildren, "I was a student of Alonzo Church".

Henkin [31] says that in 1941, and as part of the Ph.D. program, he took a logic course with Church with the following content:

1. In the first semester various systems of propositional and first-order logic were introduced, normal forms were described and established and were used in proving completeness, and the (downward) Löwenheim-Skolem theorem was discussed. In the presentation of Gödel's completeness proof[6], emphasis was given to its reductive character: the provability of a logically valid

[5] See Hodges [35], page 119.

[6] Obviously, Henkin's completeness proof was not available at the moment.

formula is reduced first to the provability of its Skolem normal form, and then to the provability of some tautology in a specified set of propositional formulas.

2. In the second semester an applied second-order system for Peano arithmetic was studied in great detail, and the Gödel's incompleteness results were derived for it; and from these followed the incompleteness of second-order logic. In connection with the incompleteness proofs, primitive-recursive functions received detailed examination in the course, but there was no time to study general recursive functions. However, their definition was mentioned, and their role in establishing the non-existence of certain decision procedures was described.

According to Henkin, although the content of the course was not in itself surprising, the style of his teacher -the way he transmitted his conception of logic- was. He used to make frequent pauses to comment that he was following the *logistic method,* to emphasize the distinction between language and metalanguage, to point out how formal language had to be defined in a completely effective way, and to explain why metalanguage (English, in his case) had to be limited. One can glimpse how his classes were conducted by reading his book **Introduction to Mathematical Logic, Vol. I.**

Doctoral theses supervised by Church. During his long life Church supervised 31 Ph.D. dissertations. These were:

Alfred Foster, Alan Turing, Stephen Kleene and Berkley Rosser in the 30's.

Enrique Bustamante, Leon Henkin and John Kemeny in the 40's.

Martin Davis, Dana Scott, Hartley Rogers, Norman Shapiro, Maurice L'Abbé, Willian Boone, Thacher Robinson, Michael Rabin, Simon Kochen, Aubert Daigneault and Raymond Smullyan in the 50's.

Peter Andrews, Robert Ritchie, James Guard, James Bennet, Robert Winder, Wayne Richter, Gustav Hensel, Willian Easton, Joel Robbin and Donald Collins in the 60's.

Richard Malitz and Anthony Anderson in the 70's.

Gary Mar in the 80's.

He supervised his last thesis in 1986, when he was more than 80 years old. (Henkin explains that the list of Church's students might be incomplete and remembers a Mexican student, Zubieta, who might have written his dissertation under Church's supervision in the 50's. This Zubieta is mentioned in the preface of Church's book, **Introduction to Mathematical Logic**; he is thanked for having written the index and for drawing Church's attention to some errors.)

Church's Personality. Everybody describes Church as a friendly and good humored man, very much balanced in his opinions. Some were surprised that a genius like him could be such a even-tempered person.

(I imagine that for many years he had to put up with comparisons with Kurt Gödel, also a professor at Princeton, but whom few would call even-tempered. Gödel died when I was at Berkeley in 1977-1978. At that time different stories were circulating about him. The most charming one -others were somewhat less charming- concerned the time when Gödel was about to obtain his American citizenship. Gödel had to have an interview with a government official, and he asked Einstein to accompany him because, it seems, he was afraid of being asked his opinion about the American Constitution, which he had read thoroughly, since he thought that if he had to give a truly sincere opinion he would have to say that it was full of contradictions.)

Because he was such a well-balanced person -both rational and reasonable- there are no funny stories about Church. He was a large man (even his handwriting was large), and, it is said, he once used this attribute during a long-winded colloquium where he was acting as moderator as a wall between a rattled speaker and the exhausted audience.

A photo of Church appears in the article dedicated to him in the **Encyclopedia of Computer Science and Engineering**. Another photo appears in Barendregt's book on lambda calculus.

One of the stories in the obituary written by Hodges recounts that in an evaluation meeting with other professors Church agreed to give a passing grade to a student with deficient work on the condition that the student would promise never to write anything about the philosophical significance of Gödel's Theorem. I would also willingly sign such an undertaking.

Church was married for 50 years, his wife dying in 1976. He had three children: Alonzo, Mary Ann and Mildred. Mary Ann is married to the well known logician John Addison who teaches at Berkeley and they live near-by in El Cerrito.

Visiting Professorships. Besides being a full professor of Philosophy and Mathematics at Princeton and Los Angeles, Church was a visiting professor at the following institutions:

1. University of Illinois (1956-57)
2. University of California at Berkeley (spring of 1960)
3. University of California, Los Angeles.(1960-61).

Honors..

- In 1967 Church was elected member of the Academy of Arts and Sciences and a member of the International Academy of the Philosophy of Science.
- In 1978 he was elected member of the National Academy of Sciences and a member of the British Academy.
- Church was awarded honorary doctorates by Case Western Reserve University (1969), Princeton (1985) and the State University of New York at Buffalo. (1990). (The day before receiving this latter degree, a symposium in his honor, organized by John Corcoran, was celebrated at the University.)

The Journal of Symbolic Logic. When the *Association for Symbolic Logic* was founded in 1935, Church became the Editor of its journal -the **Journal of Symbolic Logic-** and indeed remained a major editor for several decades. He remained in charge of the Reviews Section until 1979. This journal reached a high degree of objectivity and consistency, and was for many years *the* journal of logic.

Many people say that as a Review Editor Church was very strict and that he never allowed anyone to go too far in his/her comments. It is said that Church held that reviews could be very negative without having to be harsh; above all, he could not bear mordacity or irony, since he was convinced that these were definitive and precluded possible replies.

According to David Kaplan, Church's opinion on this matter was:

> Evaluations can be rebutted, and errors can be corrected, but sarcasm can neither be rebutted nor corrected.

It is also said that those who procrastinated in sending in their comments received *the Bomb*, a shattering letter.

Church remained au courant of publications in logic over many years and the Journal featured a wonderful, periodically updated, section on bibliography.

3. His Work.

3.1. His early writings.

Church's first paper [4] was published in 1924 and the last one [14] in 1995.

In his dissertation, Church tried to accomplish for set theory what others had done by creating non-Euclidean geometry. Lobachevsky considered the Parallel Axiom to be a postulate that could be questioned and was willing to study the axiomatic theory resulting from adding the negation of this axiom to the other Euclidean axioms. Church proposed different axioms that contradicted the Axiom of Choice as the basis for an alternative to Zermelo's set theory.

In his

- A. Church. [1926]. *Alternatives to Zermelo's assumption.* **Transaction of the American Mathematical Society**, vol. 29, pp. 178-208.

he considered as true the possibility that this axiomatization does not constitute a complete theory, even if one adds the axiom of choice, because he thought that if it were complete it should be possible to construct an F of the kind required by the axiom of choice. However, the non-constructive nature of the Axiom of Choice clearly prevented such a possibility. Some years later, when he presented his lambda calculus, he tried, unsuccessfully, to derive it in lambda calculus: it had to be taken as an axiom. In contrast with this situation, other axioms for Zermelo set theory, establishing the existence of certain sets, were not needed as such since their existence was ensured by the rules for the formation of expressions of lambda calculus.

In a paper written in 1928, clearly under the influence of his *maestro* Brouwer,

- A. Church. [1928]. *On the law of the excluded middle.* **Bulletin of the American Mathematical Society**. Vol. 34. pp. 75-78.

Church considered the possibility of eliminating the Law of the Excluded Middle. In general, his attitude towards the principles of Logic was not dogmatic, for he was open to considering the variants of systems obtained by modifying such principles. Some logicians, Langford among them, reacted strongly against Church's view, considering that logic is the system of propositions that express analytic facts of a formal nature, and that therefore there can only be one system, *the system.*

That same year (1928), the problem of the decidability of first-order logic was presented in detail and recognized as a crucial problem. A few years later both Church and Turing (apparently in an independent way) solved the problem negatively. As we shall see in section 4, it is precisely this problem, Hilbert's 10^{th} problem, which forced both of them to clarify and define the notion of effective computability.

3.2. Introduction to Mathematical Logic, vol. I.

Church's book, **Introduction to Mathematical Logic, vol. I**, is the book of logic par excellence; the book where the subject itself was defined, where the characterizing approach was made explicit, and the basic topics established for several generations of logicians. The second volume was only partially written but circulated among students in mimeographic form, as it did at Princeton University library. A previous version of the manual appeared in 1944 as a volume of the **Annals of Mathematical Studies** and as early as 1946 the manuscript was in the library collection.

This book, whose index corresponds to the syllabus of the course Henkin reports having taken [7], has an introductory chapter of 70 pages: an essay in its own right. In personal communications, Leon Henkin has told me:

> I have several times in recent decades re-read the long Introductory chapter of the book, and each time was excited by the richness of its ideas. It contains the basis for much future work by logicians, I am sure.

There, Church makes a sharp and intelligent analysis of the ontological and semantic assumptions we take for granted in classical logic. He also explains the reasons that justify the acceptance of these principles. An important part of this introductory chapter is devoted to the presentation and clarification of the mathematical and linguistic concepts forming the pillars of logical analysis: names, functions, constants, variables, and propositions. Church also studies other concepts, more logic-based; namely, connectives and quantifiers. He also comments on Frege's work, placing special emphasis on the distinction between sense and denotation. Section 07 of the introduction, specifically dedicated to the exposition of the logistic method, is of special interest and is indeed a gem. In

[7] This is not completely true, as chapter V, devoted to second-order logic, includes Henkin's proof of completenes of second-order logic.

that section Church makes explicit the principles that we necessarily accept when we are doing formal logic; principles that we now take for granted, such as the distinction between language and metalanguage, but that were very important at the time of writing since people were still emerging from of a period of confusion. He also makes clear the *effectiveness* requirements for the language and for the logic system. Such requirements are relevant:

1. When specifying primitive signs (for each sign we must be able to tell whether or not it is a primitive sign);

2. When defining formulas (for each succession of signs we must be able to tell whether or not it is a formula);

3. When specifying axioms (for each formula we must be able to tell whether or not it is an axiom);

4. When defining rules of inference (when a formula is proposed as the conclusion of an inference we must be able to tell whether or not it is correct, according to the rules).

As Church himself pointed out, these requirements allow us to say that the concept of proof is effective, to the effect that there is a method by means of which, given a finite sequence of formulas, we can determine whether it is a proof or not. He insists on the need of having an effective proof procedure since, otherwise, one can always demand a proof that the given proof is in fact a proof, and this is an infinite recurrence. Church adds that proof of a theorem in the object language must be carried out without using semantics. There are three important reasons for this[8]:

> In the first place, this demand may be considered a more precise formulation of the traditional distinction between form and matter and the principle that the validity of an argument depends only on the form -the form of a proof in a logistic system being thought of as something common to its meanings under various interpretations of the logistic system. In the second place, this represents the standard mathematical requirements of rigor that a proof must proceed purely from the axioms without use of anything (however assumedly obvious)

[8]See Church [12], page 55.

which is not stated in the axioms; but here this requirement is modified as follows: a proof must proceed purely from the axioms by the rules of inference, without use of anything not stated in the axioms or any method of inference not validated by the rules. Thirdly there is the motivation that logic systems is relatively secure and definite, as compared to interpretations which we may wish to adopt, since they are based on a portion of English as a metalanguage so elementary and restricted that its essential reliability can hardly be doubted if mathematics is to be possible at all.

Nowadays we would rewrite this as: more potential of background set theory is needed for the semantics of logic than for the calculus; accepting set theory as a basis for mathematics.

In contrast, the notion of something being a theorem is not necessarily effective, in the sense that there may be no method by means of which, given a formula, we can always determine whether or not it is a theorem.

In the book, Church insisted that we use the intuitive concept of effectiveness; that is, the concept of an algorithm we commonly use in mathematics (for instance, Euclides' algorithm to find the maximum common divisor, or Eratostenes' sieve in determining all prime numbers below a given one). A set is called decidable when we have an algorithm that tests whether or not the defining property of the set is fulfilled. When the algorithm is able to produce values of a function, we call it an effectively computable function (or rather, a computable function). Church highlights the fact that only the intuitive concept of effectiveness is needed when we wish to show that a given method is in fact effective. And in a footnote [9] he adds that the precise definition can be found in a paper by the author, giving the references of papers by Turing, Post, Kleene, Church and Hilbert & Bernays,

[9] On page 52, note 119 which I reproduce below.

For a disscussion of the question and proposal of a rigorous definition see a paper by the present writer in the **American Journal of Mathematics**, vol. 58 (1936), pp.345-363, especially § 7 thereof. The notion of effectiveness may also be described by saying that an effective method of computation, or algorithm, is one for which it would be possible to build a computing machine. This idea is developed into a rigorous definition by A. M. Turing in the **Proceedings of the London Mathematical Society**, vol. 42 (1936-1937), pp. 230-265 (and vol. 43 (1937), pp. 544-546). See further: S. C. Kleene in the **Mathematische Annalen**, vol. 112 (1936), pp. 727-742; E. L. Post in the **Journal of Symbolic Logic**, vol. 1 (1936), 103-105; A. M. Turing in the **Journal of Symbolic Logic**, vol. 2 (1937), pp. 153-163; Hilbert and Bernays, **Grundlagen der Mathematik**, vol. 2 (1939), Supplement II.

where alternatives definitions are proposed and the equivalences between them are proved.

The intuitive description of the concept of effective procedure (or equivalently, of effective computability) is sufficient if one wishes to establish that a function is in fact computable. In the book Church shows that propositional logic is decidable (Section 15: `Tautologies, the decision problem`) by simply making clear that the truth tables themselves constitute a decision procedure [10]. Undecidability is not proved for first-order logic; Church proves, however, that there are certain interesting decidable sets of first-order formulas such as unary predicate formulas [11]. In fact, undecidability is beyond the scope of the book because no mathematical definition of the concept of an effective procedure or algorithm is provided in it. Thus, it is impossible to prove that a set is undecidable or that a function is not computable.

It is curious that in the book there is no presentation of any of Church's three major contributions to Logic and Computer Science: lambda calculus, the thesis that bears his name, and the proof of the undecidability of first-order Logic. I believe the decision was taken on the basis of sound pedagogical considerations.

One cannot conclude these comments without mentioning the juicy historical sections and the footnotes. These footnotes underscore the attention Church gave to details, his scholarly excellence and erudition, and also how and to what extent his book was a product of a palimpsest of rewritings. Today, with the availability of text processors, the traces of previous versions of a work disappear, and with publishers's dislike of footnotes (they usually request of authors that, at most, they put them at the end), such notes are all but disappearing. For all its notes, I cherish Church's book, especially the very long Introduction, and I agree with Henkin that one can still find in it material for future work, especially for those interested in a rigorous Philosophy of Logic, taking into account the Fregean tradition.

[10] This is basically Post's proof [48].

[11] Decidability of unary predicate logic, both for first-order logic and second-order logic, is a result of Löwenheim [45].

3.3. Beauty and the Beast (Logic and Computer Science).

When I say that Church must have felt directly responsible for the development of the theoretical basis of Computer Science, I am thinking of the three contributions mentioned above; all three were obtained in the 30's and were to have a decisive impact on this discipline. As we shall see, these contributions are directly connected to the concept of computability, which is evidently the central concept of Computer Science.

Computer Science. Computer Science is concerned with computers and computation. Computer Science is interested in the invention and analysis of algorithms, the design of programs and programming languages to express algorithms, and the construction of systems of computation, including machines, to execute or implement such programming languages.

Logic. Logic is thought to have as its principal objective the analysis and codification of reasoning; it is interested in the meaning and transformation of the statements we use in our discourse about the world.

In logic, an artificial language is defined with precision, meaning is attributed to the statements of the language by means of mathematical models, and a calculus is defined; *i.e.*, a system of axioms and rules governing the sound transformations of the statements of the formal language. Depending on the level of abstraction required and on the area of reality one has to deal with, there are different types of Logic.

Furthermore, it has been within mathematical logic that a theory of computation has been developed, that the mathematical concept of recursion has been defined, and that the power of algorithms has been characterized mathematically. This was done during the 30's, before the arrival of computers, although if had it not been done then, it would have been implemented later on in computer science.

Logic within Computer Science. Logic plays an essential role in all levels of description of the workings of a computer, as well as in the execution of a program in a machine. It plays a role at basic level, governing the logical design of the hardware; it plays a role in the cascade of languages that pass from the user to the machine language (the one the machine understands); it also plays a role when, working at metatheoretical level, we reason about programs; that is to say, when we do logic of programs.

Church's contributions have direct bearing on all these levels. He not only defines the mathematical concept of recursiveness, but also provides a programming language and establishes, in a clear fashion, the frontier of computability. I believe that an important part of these results must be attributed to the admirably suited properties of the λ language and calculus, which I briefly present in the appendix.

4. Some of Church's Miracles.

In what follows I shall endeavour to render plausible my thesis that Church's great discovery was, undoubtedly, the lambda calculus, and that his other important contributions, as well as some of his disciples, are fruit of this initial successful and sensible choice. The argumental trend will be as follows:

(1) The lambda calculus. In the early decades of this century it was possibly considered unavoidable that a logician proud of his/her work should present his/her own system of logic. Church created the lambda calculus using function as the basic concept and made a clear distinction between the value of a function for a given argument, $F(x)$, and the function itself, $\lambda x F(x)$. To better illustrate this, let us examine one of Church's own examples [12]:

> If we say, $(x^2 + x)$ *is greater than 1,000* we make a statement which depends on x and actually has no meaning unless x is determined as some particular natural number. On the other hand, if we say (x^2+x) *is a primitive recursion function*, we make a definite statement whose meaning in no way depends on a determination of the variable x (so that in this case x plays the role of an apparent, or bound variable). The difference between the two cases is that in the first case the expression (x^2+x) serves as an ambiguous, or variable, denotation of a natural number, while in the second case it serves as the denotation of a particular function. We shall hereafter distinguish by using (x^2+x) when we intend an ambiguous denotation of a natural number, but $(\lambda x(x^2 + x))$ as the denotation of the corresponding function - and likewise in other cases.

It was a success indeed to have chosen functions as the basic building blocks, but to distinguish the values of the function from the function itself in the language

[12] See [10]. (Page 6 of the 1951 edition.)

was an even better achievement. If one adds to these the characteristic nesting properties allowed in formalization, the language is now ready to express recursion. In fact, in contrast with other axiomatic foundational systems, in lambda calculus the existence of mathematical objects (functions in our case) does not need to be postulated by axioms since they can be generated by using the formatting expressions rules of the calculus itself.

This language is so compact and so suited to expressing functionality, that it becomes a programming language. As I mentioned previously, inspiration for the LISP, developed by John Mc Carthy, can be attributed to lambda calculus.

Although lambda calculus was circulating for a number of years, it only appeared in print in the 40's [13]. At the beginning of the 30's Church was already aware that both the successor function and the addition were $\lambda-$definable, but it was his student Kleene who gradually discovered how wide the class of definable functions was. These results were the basis of his Ph.D. thesis; to a certain extent the origin of recursion theory. However, the results of his investigations drove him further than expected. Let us see what Martin Davis [19] says in this respect:

> Church published a pair of substantial papers on the system he developed and set his students Stephen C. Kleene and J. Barkley Rosser to work on it. Their work was extremely effective, if not exactly what one dreams of having one's graduate students accomplish for one: Kleene and Rosser proved that Church's system was inconsistent!

Despite this initial slip, it was soon realized that lambda definability was equivalent to recursiveness and to Turing computability, thus providing a strong argument supporting the so-called Church thesis; that is, to take either as a possible definition of effective computability or algorithm. Additionally, using his thesis Church was able to prove undecidability for first-order logic.

We shall later see how lambda calculus was used as a language for type theory, thus producing a wonderful effect: type theory in this presentation is extremely elegant. Moreover, this type theory presentation allowed Henkin to prove his completeness theorem in the way everyone today proves in any logic, not only in typed lambda calculus.

[13] The small volume [10] is rather interesting. An updated presentation can be found in the book by Hindley [33] and in Barendregt [2].

(2) The Entscheidungsproblem. One of the most striking problems in the thirties was decidability. Hilbert and Ackerman [34] posed it as follows:

> The Entscheidungsproblem is solved if one knows a procedure which will permit one to decide, using a finite number of operations, on the validity, respectively the satisfiability, of a given logical expression.

This problem was of capital importance for Hilbert's school, which at that time expected a positive solution for the very famous Hilbert tenth problem. Pretty soon, however, the atmosphere turned to skepticism and a negative solution was expected, since otherwise certain unlikely consequences seemed to emerge. In particular, after Gödel's incompleteness results the scenario changed and it was almost inconceivable that the Entscheidungsproblem could be decidable[14].

As mentioned above, to show that a set is decidable only requires one to pinpoint the generating algorithm. Since the intuitive notion of effectiveness is sufficient to identify an effective procedure, this can be done without a mathematical definition of computability. Nevertheless, to show undecidability the intuitive notion does not suffice and a mathematical definition is a must.

Thus, from the desire to solve undecidability problems arose the need for a mathematical definition of computability. In particular, solving the famous Hilbert tenth problem could have been the incentive in Church's case, as he worked to find an acceptable definition of computability.[15]. Since the investigations of his student Kleene and his own had revealed the great expressive power of the language introduced by Church, it seemed reasonable to assume that such a definition could possibly be $\lambda-$definability.

As regards the *Entscheidungsproblem*, Church proved that for first-order logic there is no algorithm for validity. The paper where Church proves this theorem [8] was published in 1936 and is based on the results of a previous paper by the author [7] published the same year; the recently introduced technique of Gödelization [16]

[14] Gandy [24] very elegantly describes what the situation was at the time.

[15] We know that this was also the case for Turing; he defined computability by an abstract machine (*Turing machines).*

[16] It is also said that if Gödel had not obtained his incompleteness results, they could have been derived from the results of Church; let us see what Shoenfield [51] says in a footnote about this:

"Kleene has pointed out that if Gödel had not already proved his Incompleteness Theorem, Church might well have derived it from his results. The reader interested in such a derivation of is also employed. (At the end of this paper I have added a proof of undecidability for first-order logic; the line of reasoning is Church's, the presentation is the one commonly used today.)

(3) Computability: recursiveness. Classical recursion theory comprehends the study of functions defined over the natural numbers, but at present recursion theory has developed its own potential and methods, and it has reached a stage where the abstract development is only matched by its unforeseen applications. Pioneering work on recursion theory was done by Dedekind, when in 1888 introduces the study of the functions over the natural numbers defined by equations and by the induction principle, as defined and made precise by himself.

As far a its present stage is concerned, whose range covers all effectively computable functions, the origins have to be investigated at Princeton; it certainly begins with Church, but whole parenthood should be attributed to Kleene. He defined the subject matter, drove it and draw the boundaries. Normal form theorem and Recursion theorem are his.[17]

As for the definition itself, we have mentioned above that several versions of the concept were circulating, although as we will see in next section there was a certain resistance to taking any of them as the actual definition. Several of these concepts appeared in the 30's to characterize notions that seemed, in principle, to be different: the first one was Gödel's characterization of the functions defined by means of recursion; the second was the concept of function defined by the λ-operator, which Church and Kleene introduced, and the third one was the notion of a function computable by an abstract machine -*i.e.*, Turing machines. It was soon proved that all three notions define the same functions. Although it was quite clear that all functions defined by any of the previously mentioned methods is effectively computable, this result was of no help in proving that a set is undecidable;rather the converse is needed. It belongs to its own essence that a statement like that could not be proved, but if one insists on not accepting it as a definition, it demands a new status.

the Incompleteness Theorem can consult my book **Mathematical Logic.** Of course, this takes nothing away from Gödel's supreme achievement in discovering the Incompleteness Theorem".

[17] Davis [18] is an antology of basic papers on the subject ía de los artículos básicos de este campo. In Kleene [43] one find a historical study of the early times.

(4) Church's thesis. Another of Church's most remarkable merits is to have formulated the thesis that the precise mathematical concept of a recursive function, as defined by Turing, corresponds exactly to the intuitive concept of effective computability.

Shoenfield [51] says that Church's thesis arose in a accidental way and I would tent to say something similar; that it emerged in quite a natural way. Having introduced his lambda system, and being interested in the scope of lambda definability, Church gradually realized (as I mentioned above, partially due to his student Kleene) how wide this class actually was. It was clear that all $\lambda-$definable functions were computable and it seemed reasonable to assume that all computable functions were $\lambda-$definable. That is, that the intuitive notion of computability (or effectiveness) corresponds exactly to $\lambda-$definability. As early on as 1933, Church formulated this identification in the form of a definition. Afterwards, Kleene restated Church's definition as a thesis and it has remained in this form until today. The first formulation of the proposal was to identify effective computability with $\lambda-$definability; this was publicly announced as recursiveness and later widely accepted as computability by a Turing machine. The vicissitudes of the thesis can be followed in Davis [19].

Being a thesis, there is no mathematical proof for it, although there are strong reasons for accepting it:

(1) The first takes as its argument the experience that mathematicians have stored for centuries: all known algorithms are recursive. In this sense, Gandy [24] adds:

> Kleene's dissertation shows, as we might now put it, the power of the $\lambda-$calculus as a high-level programming language. A $\lambda-$term can take other $\lambda-$terms (subroutines), instead of numbers, as inputs; and, if taken in the right order, successive conversions do correspond to a natural way of implementing the program which the $\lambda-$term represents. Of course Kleene did not think in these terms, but this way of putting it helps one to appreciate the intuitions which led Church and Kleene to believe in the truth of the thesis.

(2) The second argument in favor of the thesis takes the equivalence among the three concepts introduced to characterize what an algorithm actually is as a very strong reason: It is impressive and indeed rather soothing that from such different starting points one can define exactly the same functions.

(3) The third argument, which is the one that undoubtedly convinced Gödel, has to do with the kind of analysis Turing makes of the concept of computability; he is not so much interested in what a computable function is but rather in what it does; he is more interested in the process than in the nature of the functions. Turing started his analysis by saying:

> The real question at issue is *What are the possible processes which can be carried out in computing a number?*

Functions are built from very elementary functions, which are clearly algorithm-like, by means of processes that raise no doubts as to the algorithmc character.

(4) In fact, this argument is very similar to one of Church's arguments, which Gandy [24] explains thus:

> *The Step-by step Argument.* This is Church's chief argument. He considers the evaluation of a value fm of a function either by the application of an algorithm, or by the derivation of $fm = n$ from axioms about f in some formal system. In each case the evaluation proceeds in a series of steps. If each step is recursive then f will be recursive. As evidence for the premise he points out that Gödel had shown that the steps in a proof in Gödel system P are primitive recursive.

(5) Finally, we have the reason that convinced Kleene: this was the failure of the diagonal argument. We use the term diagonalization to refer to the process that mimics Cantor's argument to show that $\wp\omega$ is not bijectable with ω. This process allows one to point out sets (in the case mentioned, $Y = \{x \mid x \notin f(x)\}$) that have been left out from a supposedly complete enumeration of a given domain..

Tt was now possible now to prove undecidability for a set, since it sufficed to show that the function of membership to that set was not a recursive function. Church did this for a problem related to lambda calculus in 1936. The same year, using Gödel's results, he proved that if first-order logic were decidable, the system proved undecidable in his previous paper, would be decidable.

(5) Lambda calculus and type theory. Firstly, Church's $\lambda-$calculus is a useful tool for investigating the nature of mathematical operations. And we have seen that important results are associated with $\lambda-$definability. Secondly, and this is our concern now, lambda calculus can be used as a basis to present type theory.

Given that $\lambda-$calculus gives rise to inconsistencies, Church formulated the type theory that Russell and Withehead introduced in the Principia in the powerful lambda language. A calculus for simple type theory was also defined, taking some of the lambda conversion rules. The offshoot of this was fantastic, since added to the formalizing capacity of the lambda language was the naturalness of type representation. (To me, the solution proposed in type theory to solve paradoxes seems quite sensible; namely, to distinguish levels or types of variables. The reason is that when one wishes to describe a mountainous landscape, such as we like to imagine that the hierarchy of sets or of types represents, a flat language of representation is useless; one need a language of representation that will include "contour lines".

Church [9] proposed a simple typed $\lambda-$language in which the so-called logical or mathematical paradoxes, such as Russell's or Cantor's, disappear. The reason for this must be sought in the syntax of the language, where the autorreflexive constructions causing paradoxes are avoided. The so-called semantic paradoxes, such as the Liar paradox, are avoided when Tarki's semantics is adopted for the purpose of clearly distinguishing object language from metalanguage. By doing this, Church avoids the additional complication of establishing order, as Russell does in the ramified theory of types. (In appendix, I present the language of Church for the simple theory of types [18].)

Many years later Church published a paper [13] in which he compared Russell's and Tarski's solutions to paradoxes.

Another use of $\lambda-$calculus was made by Church [11] in the fifties. In this paper Church accepted Frege's thesis, according to which in order to understand a language it is not enough to know the structure of the symbolic language and the universe of objects to which it refers but rather one needs to place a third level of abstract entities; namely, senses or concepts. In that paper Church develops a mathematical theory of senses understood thus and of their relation to the objects

[18] In Manzano [46] this lambda type theory is presented, as well as a second-order language also containing the lambda abstractor. The book of Andrews [1] also includes some chapters on type theory.

of the universe. To do this, he uses the language of types of his 1940 article and adds a hierarchy of types for the senses.

(5) Henkin's completeness. When a formal language is introduced, together with an associated deductive calculus and a semantics, it is mandatory to investigate the adecuacy of these two procedures for selecting formulas. Gödel had solved the issue for first-order logic positively, and negatively for any logical system in which arithmetic could be embedded.

The lambda calculus for the theory of types, coupled with the usual semantics over a standard hierarchy of types, is strong enough to formalize arithmetic. However, it is obviously incomplete.

Nevertheless, Henkin proved that if the formulas are interpreted in a less rigid way, accepting other hierarchies of types that do not necessarily contain all the functions but do contain at least the definable ones, one can easily show that all the consequences of a set of hypotheses are provable in the calculus. The valid formulas in this new semantics, named general semantics, are reduced to coincide with the formulas generated by the rules of the calculus.

It is well known that the completeness proof of Henkin rests on the proof that every consistent set of formulas has a model. To build the model, one previously extends the consistent set of formulas to a maximal consistent set of formulas, containing witnesses, and the model constructed is the model described by the formulas in the maximal consistent set since these sets are characterized as being a very precise and detailed description of a model.

Naturally, I will not enter here into a detailed examination of Henkin's theorem; I am only interested in showing how its discovery is related to lambda calculus.

Henkin explains that he was trying to represent the functions that can be named by $\lambda-$expressions to himself mentally. Clearly, if we have a countably infinite universe, the universe of sets of individuals will be uncountable. However, given that $\lambda-$language is only countable, the set of sets that can be named can only be countable too. It seems that the definitive *click* occurred when Henkin realized that in trying to represent the objects of this hierarchy of types, in order to avoid repetitions he was using the rules of lambda conversion and that the syntax and calculus were therefore involved in the description of the semantic objects. Henkin used the calculus as a criterion to identify the named objects by means of M^α and N^α; especially the claim that $\vdash M^\alpha = N^\alpha$. He finally completed the proof when he realized that in order to reduce the universe of objects named by means of propositions to only two objects, it would be necessary to expand the axioms until they constituted a maximally consistent set, so that the number equivalence classes induced by the relation $\vdash M^\alpha = N^\alpha$ would be reduced to two.

I shall quote some of the clue phrases from the beautiful and illustrative tale Henkin [31] himself made of his discovery.

> I decided to try to see just which objects of the hierarchy of types did have names in $\mathfrak{T}$.
>
> ...
>
> As I struggled to see the action of functions more clearly in this way, I was struck by the realization that I have used $\lambda-$conversion, one of the formal rules of inference in Church's deductive system for the language of the theory $\mathfrak{T}$. All my efforts had been directed toward *interpretations* of the formal language, and now my attention was suddenly drawn to the fact that these were related to the *formal deductive system* for that language.
>
> ...
>
> As soon as I observed this, it occurred to me that if we were to add further cwffs of type 0 to the list of formal axioms, this would have the effect of reducing the number of elements in $\mathfrak{D}'_0$ and that ultimately, by taking a maximal consistent set of axioms, the number of elements in $\mathfrak{D}'_0$ would be two.

5. Appendix.

5.1. Type Theory.

In what follows I shall briefly present the simple theory of types as can be found in Church's 1940 paper; for the semantics of type theory, readers are referred to two papers by Henkin ([27] and [30]).

The Hierarchy of Types. The theory of types presented there, that I shall call $\mathfrak{T}$,could be described as follows: Types are structured in a hierarchy. The basic types are:

- $\mathfrak{D}_1$, a non-empty set, the set of the individuals of the hierarchy; and

- $\mathfrak{D}_0$, the domain of truth values. (Since we are working in a binary logic, such values are limited to T and F)

- The rest of the domains are built from these basic ones as follows: Given constructed domains $\mathfrak{D}_\alpha$ and $\mathfrak{D}_\beta$, we define $\mathfrak{D}_{(\alpha\beta)}$ as the domain formed by all the functions from $\mathfrak{D}_\beta$ to $\mathfrak{D}_\alpha$. Sets are represented by means of their characteristic functions, and thus $\mathfrak{D}_{(0\beta)}$ represents the power set of $\mathfrak{D}_\beta$, whose elements are all the subsets of $\mathfrak{D}_\beta$.

In order to talk about this hierarchy a formal language is introduced. The one in Church's paper is, as we have said, a language of types that has different kinds of variables corresponding to the different types. The symbol λ is available in the language and can be used with variables of different types to form abstractors, as explained below.

Language

Alphabet. This is composed of variables of different types, each with a superscript indicating the type.

In this paper, Church also introduces the constants $N_{(00)}$ and $A_{((00)0)}$ (for negation and conjunction) and $\Pi_{(0(0\alpha))}$ (that will allow us to express quantification). Finally $\iota_{(\alpha(0\alpha))}$ is a selector operator.

Expressions There are two fundamental ways of forming expressions:

- Simple expressions: Each constant or variable is a formula whose type is that of its index.

- Complex expressions are of two kinds: (1) Complex expressions with λ . If X^β is a variable and M^α is an expression, then $(\lambda X^\beta M^\alpha)$ is an expression of type $(\alpha\beta)$. (2) If $F^{(\alpha\beta)}$ and B^β are expressions, $F^{(\alpha\beta)}B^\beta$ is an expression of type α.

Interpretation The interpretation of these expressions is the following:

- Variables take their values in the corresponding universes or domains. Constants have values determined in advance: for instance, $N_{(00)}$ is negation and $A_{((00)0)}$ is disjunction.
- The expression $(\lambda X^{\beta} M^{\alpha})$ is interpreted as a function from $\mathfrak{D}_{\beta}$ to $\mathfrak{D}_{\alpha}$. (For instance, $(\lambda X^{\beta} X^{\beta})$ is the identity function on the set $\mathfrak{D}_{\beta}$. However, $(\lambda X^{\beta}\phi)$ is interpreted as the class of elements of $\mathfrak{D}_{\beta}$ satisfying the condition expressed by the formula ϕ.)
- Finally, the expression $F^{(\alpha\beta)}B^{\beta}$ is interpreted as the value of the function interpreting $F^{(\alpha\beta)}$ for the value that B^{β} denotes.
- The function $\Pi_{(0(0\alpha))}$ assigns to each subset of $\mathfrak{D}_{\alpha}$ (which is an element of $\mathfrak{D}_{(0\alpha)}$) the value F except when this subset is $\mathfrak{D}_{\alpha}$ itself -i.e., the totality. Hence, we can think that this predicate of predicates asserts their universality. It can then be understood that $\Pi_{(0(0\alpha))}$ $(\lambda X^{\alpha}\phi)$ expresses quantification, $\forall X^{\alpha}\phi$. Literally it says: the class of individuals satisfying ϕ is universal; i.e., it is the whole universe $\mathfrak{D}_{\alpha}$. (There is a language of set theory that I like even better; it has λ and identity as the only primitive symbols. For example, $\forall X^{\alpha}\phi$ is defined by the formula: $(\lambda X^{\alpha}\phi) = (\lambda X^{\alpha} X^{\alpha} = X^{\alpha})$.[19])
- The interpretation for $\iota_{(\alpha(0\alpha))}$ is that of a selector that assigns an element of $\mathfrak{D}_{\alpha}$ to each element of $\mathfrak{D}_{(0\alpha)}$ (that is, to each subset of $\mathfrak{D}_{\alpha}$). When the interpretation of the formula $(\lambda X^{\beta}\phi)$ is a singleton, the selector assigns to it its unique element, and hence $(\iota_{(\alpha(0\alpha))}(\lambda X^{\alpha}\phi))$ is abbreviated as $(\iota X^{\alpha}\phi)$, and is taken as a description operator.

The difference in interpretation between the constants $N_{(00)}$, $A_{((00)0)}$ and $\Pi_{(0(0\alpha))}$ on the one hand, and $\iota_{(\alpha(0\alpha))}$, on the other, relies on the fact that in the first case the interpretation is fixed in advance and in a unique way. In the second case, one only requires that it be any choice function of the adequate type, although no function is fixed in advance. It seems that this asymmetry, or indetermination, lies at the roots of the research that led Henkin to discover his completeness proofs. Thus, this asymmetry has provided a key part to all logic, the king of proofs of formal systems, since its procedure, unlike the one used by Gödel to demonstrate completeness for first-order logics, is so versatile that it can be adapted to other logics.

[19] See Henkin [29] and [30], also see Andrews [1].

Deductive Calculus. The deductive calculus offered by Church in the above-mentioned paper includes six rules of inference and eleven axioms. The first three rules are substitution rules, including the rules of λ-conversion; among them the following one can be found:

$\frac{(\lambda X^{\beta} M^{\alpha}) N^{\beta}}{S^{N\beta}_{X\beta}(M^{\alpha})}$ (one application of this rule allows us to pass from $(\lambda X^{\alpha} R X^{\alpha})A$ to RA.)

and its converse.

There is also an instantiation rule, a modus ponens and a generalization rule.

Besides having the propositional axioms and the axioms for quantification, the following axiom for descriptions is included:

$(f^{(0\alpha)} X^{\alpha}) \rightarrow (\forall Y^{\alpha}(f^{(0\alpha)} Y^{\alpha} \rightarrow X^{\alpha} = Y^{\alpha}) \rightarrow f^{(0\alpha)}(\iota_{(\alpha(0\alpha))} f^{(0\alpha)}))$

Church also adds extensionality, choice, and infinity since he wanted his calculus to serve as an alternative to set theory -i.e., as a foundational system. Comprehension does not need to be added as axiom, since it follows from the lambda conversion rules of the calculus. Thus, we have comprehension as a logical theorem in Church's system.

$(f^{(0\alpha)} X^{\alpha}) \rightarrow f^{(0\alpha)}(\iota_{(\alpha(0\alpha))} f^{(0\alpha)}))$ is the elegant formulation of the axiom of choice that Church proposes in his language.

$\forall X^{\beta}(f^{(\alpha\beta)} X^{\beta} = g^{(\alpha\beta)} X^{\beta}) \rightarrow f^{(\alpha\beta)} = g^{(\alpha\beta)}$ is extensionality.

Henkin[31] comments on this as follows:

> Several features of the theory $\mathfrak{T}$ sketched above were interesting to me, but I was especially attracted by the neatness and shortness of the formula expressing the axiom of choice. It seemed to me that the symbol $\iota_{\alpha(0\alpha)}$ was put into the formal language of $\mathfrak{T}$ originally to serve the function of the definite article *the,* as expressed in Axiom 9^{α}(axiom for descriptions), and that its availability to provide such a succinct formulation of the axiom of choice was a fortuitous circumstance that must have come to Church as an inspired afterthought.

Natural numbers are also defined in this paper, following Russell and Frege, and Peano's postulates are proven for them as theorems.

In particular, zero, one and two are defined as follows:

$0^{\alpha'} = \lambda f^{(\alpha\alpha)} \lambda X^{\alpha} X^{\alpha}$

$1^{\alpha'} = \lambda f^{(\alpha\alpha)} \lambda X^{\alpha} (f^{(\alpha\alpha)} X^{\alpha})$

$2^{\alpha'} = \lambda f^{(\alpha\alpha)} \lambda X^{\alpha} (f^{(\alpha\alpha)} (f^{(\alpha\alpha)} X^{\alpha}))$

The successor function and the concept of a natural number are also defined using lambda expressions. In this paper by Church the predecessor function, as defined by Kleene [37], is also included. It is believed that the definability of this function helped both Church and Kleene to foresee the strong power of the notion of $\lambda-$definability.

5.2. Undecidability of First-Order Logic.

Undecidability for first-order logic is usually proved indirectly: one takes an undecidable but finitely axiomatizable system and proves that first-order decidability implies the decidability of that system. Church [7] takes the system introduced in his previous paper [8]; we now use system **Q** (Robinson's Arithmetic): This theory only contains seven axioms; namely:

$$\mathbf{Q}_1 \equiv \forall xy(\sigma x = \sigma y \rightarrow x = y)$$
$$\mathbf{Q}_2 \equiv \forall x(c \neq \sigma x)$$
$$\mathbf{Q}_3 \equiv \forall x(x \neq c \rightarrow \exists y(x = \sigma y)$$
$$\mathbf{Q}_4 \equiv \forall x(x + c = x)$$
$$\mathbf{Q}_5 \equiv \forall xy(x + \sigma y = \sigma(x + y))$$
$$\mathbf{Q}_6 \equiv \forall x(x \times c = c)$$
$$\mathbf{Q}_7 \equiv \forall xy(x \times \sigma y = (x \times y) + x)$$

Gödel's well known machinery allows us to reduce the decidability of a set to the recursiveness of the set of its Gödel numbers. Thus, a theory is recursive if and only if the set of Gödel numbers of the theorems in the theory is recursive; or equivalently, its characteristic function is. Given a decidable theory, there is an effective way to decide whether or not a sentence is a theorem: one finds its Gödel number first and then uses that number as an argument in the characteristic function. The sentence is a theorem if and only if the value is 1. Conversely, if the theory is not decidable and Church's thesis is taken for granted, there is no effective method to decide whether a sentence is a theorem. If it were a theorem, the characteristic function of Gödel's numbers of theorems would be effectively computable, and thus recursive, using Church's thesis.

To prove undecidability for **Q** [20], one uses Gödel's tools (i.e., gödelization and diagonalization) and some properties of **Q** (in particular, that all recursive functions are representable in **Q**).

The line of reasoning can be presented as follows:

(1) Let $\theta \equiv \wedge_{1\leq i\leq 7}\mathbf{Q}_i$ (The conjunction of all axioms in **Q**).

(2) By the deduction theorem we have: $\mathbf{Q}\vdash \varphi \Longleftrightarrow \vdash \theta \rightarrow \varphi$ (Using completeness and soundness we can paraphrase this result by saying: a validity test is equivalent to a test of membership to the theory **Q.)**. In particular, to decide whether φ is a logical theorem, we must test the validity for $\theta \rightarrow \varphi$.)

(3) Let $\#\alpha$ be the Gödel number of the formula α. Let f be a function mapping each natural number n to $\#(\theta \rightarrow \delta_n)$, where, $\#\delta_n = n$. Obviously, this function is recursive. For each formula φ, whose Gödel number is n, $f(n) = \#(\theta \rightarrow \varphi)$.

(4) Let $\mathfrak{P}_0 = \{\#\gamma \mid\vdash \gamma\}$; i.e., the set of all Gödel numbers of logical theorems in first-order logic.

(5) Let $\mathfrak{P} = \{\#\varphi \mid \mathbf{Q} \vdash \varphi\}$. Thus, $\mathfrak{P} = \{\#\varphi \mid\vdash \theta \rightarrow \varphi\} = \{n \mid f(n) \in \mathfrak{P}_0\}$. But $\mathfrak{P}$ is not recursive, since this is the set of Gödel numbers of the theorems of **Q**, and we already know that **Q** is undecidable.

(6) Therefore, $\mathfrak{P}_0$ is not recursive either .

(7) The conclusion is that first-order logic is undecidable; there is no validity test.

Remark 1. *In 1985, in a Congress of the ASL, Robin Gandy*[21]*, who was a student of Turing and, it is said, the last person he saw in life, gave a talk where he asked himself why everything had happened in 1936? Since then I have gained a better understanding of his reasons for asking such a question.*

[20] A proof of this theorem can be found in Boolos [3], page 175. The proof is based on results obtained in the book. Unfortunately, this great logician also died recently, in the spring of 1996.

[21] Gandy also sadly died in 1995. The talk was later reflected in a magnificent book.[24].

References

[1] Andrews, P. B. [1986]. **An Introduction to Mathematical Logic and Type Theory: To Truth through Proof.** Orlando, Fla: Academic Press.

[2] Barendregt, H. P. [1981]. **The Lambda Calculus, its Syntax and Semantics.** Amsterdam: North Holland Publishing Company.

[3] Boolos, G. S. y Jeffrey, R. C. [1992]. **Computability and Logic.** Cambridge University Press. (revised and enlarged edition; the first edition is from 1974).

[4] Church, A. [1924]. *Uniqueness of the Lorentz transformation.* **American mathematical monthly.**

[5] —[1926]. *Alternatives to Zermelo's assumption.* **Transaction of the American Mathematical Society**, vol. 29, pp. 178-208.

[6] —[1928]. *On the law of excluded middle.* **Bulletin of the American Mathematical Society.** vol. 34. pp. 75-78.

[7] —[1936]. *An unsolvable problem of elementary number theory.* **American journal of mathematics.** vol. 58

[8] —[1936]. *A note on the Entscheidungsproblem.* **The Journal of Symbolic Logic.** vol. 1, Number 1. Marzo de 1936. pp 40-41

[9] —[1940]. *A formulation of the simple theory of types.* **The Journal of Symbolic Logic.** vol. 5. pp. 56-68.

[10] —[1941]. *The calculi of lambda conversion.* **Annals of Mathematical Studies**. number 6, 1941. Princeton University Press- Oxford University Press.

[11] —[1951]. *A formulation of the logic of sense and denotation.* **Structure, Methods and Meaning**, Essays in Honor of Henry M. Sheffer, New York. pp. 3-24. (Revised in NOUS: vol. 7 (1973), pp. 24-33; vol. 8 (1974), pp. 135-156; and vol. 27 (1993), pp. 141-157.)

[12] —[1956]. **Introduction to Mathematical Logic.** vol I. Princeton: Princeton University Press.

[13] —[1976] *Comparison of Russell's resolution of the semantical antinomies with that of Tarski.* **The Journal of Symbolic Logic.** vol. 41. núm. 4.

[14] —[1995]. *A theory of the meaning of names.* **The heritage of Kazimierz Ajdukiewicz.** Rodopi

[15] Church, A. y Kleene, S. C. [1936]. *Formal definitions in the theory of ordinal numbers.* **Fundamenta Mathematica.** vol. 28. pp. 11-21.

[16] Church, A. y Quine, W. V. [1952]. *Some theorems on definability and decidability.* **The Journal of Symbolic Logic.** vol. 17. núm. 3..

[17] van Dalen, D. [1983]. *Algorithms and Decision Problems: A Crash Course in Recursion Theory.* (en Gabbay y Guenthner 1983)

[18] Davis, M. [1965]. **The Undecidable.** Rave Press. Nueva York.

[19] —[1982]. *Why Gödel Didn't Have Church's Thesis.* **Information and Control,** 54, pp. 3-24.

[20] —[1988]. *Mathematical Logic and the origins of modern computers.* (in Herken [32]).

[21] —[1995]. *American Logic in the 1920s.* **Bulletin of Symbolic Logic**, vol. 1, Number 3, Sept. 1995.

[22] Feferman, S. [1988]. *Turing in the Land of O(z).* (in Herken [32]).

[23] Gabbay, D. y Guenthner, F. [1983]. **Handbook of Philosophical Logic.** vol. I. Dordrecht: Reidel Publishing Company.

[24] Gandy, R. [1988] . *The Confluence of Ideas in 1936.* (en Herken [32]).

[25] Heijenoort, J. ed. [1967]. **From Frege to Gödel.** Harvard: Harvard University Press.

[26] Henkin, L. [1949]. *The completeness of the first order functional calculus.* **The Journal of Symbolic Logic.** vol. 14, pp. 159-166.

[27] —[1950]. *Completeness in the theory of types.* **The Journal of Symbolic Logic.** vol. 15. pp. 81-91.

[28] —[1953]. *Banishing the rule of functional variables.* **The American Mathematical Monthly.** vol. 67. núm. 4.

[29] —[1963]. *A theory of propositional types.* **Fundamenta Mathematicae.** vol. 52. pp. 323-344.

[30] —[1975]. *Identity as a logical primitive.* **Philosophia.** vol. 5, pp. 31-45.

[31] —[1996]. *The discovery of my completeness proofs*, Dedicated to my teacher, Alonzo Church, in his 91^{st} year, **Bulletin of Symbolic Logic**, vol. 2, Number 2, June 1996. (presented on August the 24^{th}, 1993 at the XIX International Congress of History of Science, Zaragoza, Spain).

[32] Herken, R. ed. [1988]. **The Universal Turing Machine: A Half-Century Survey.** Oxford: Oxford University Press.

[33] Hindley, R. y Seldin, J. [1986]. **Introduction to combinators and $\lambda-$.calculus.** Cambridge: Cambridge University Press.

[34] Hilbert, D. y Ackermann, W. [1928]. **Grundzüge der theoretischen Logik.** Springer-Verlag. Berlín.

[35] Hodges, A. [1983]. **Alan Turing: The Enigma.** Published by Burnett Books Ltd. I am using the 1992 edition, by Vintage. London.

[36] Horn, B. K P. and Winston, P. H. [1981]. **LISP.** Reading: Addison-Wesley Publishing Company.

[37] Kleene, S. C. [1935]. *A theory of positive integers in formal logic.* **American journal of mathematics.** vol. 57. pp. 153-173, 210-244.

[38] —[1936a]. *General recursive functions on natural numbers.* **Mathematische Annalen**, vol. 112 (1936), pp. 727-742.(en [18])

[39] —[1936b]. *$\lambda-$definability and recursiveness.* **Duke Mathematical Journal.** vol. 2. pp. 340-353.

[40] —[1936c]. *A note on recursive functions.* **Bulletin of the American Mathematical Society.** vol. 42. pp. 544-546.

[41] —[1952]. **Introduction to metamathematics.** Amsterdam: North Holland Publishing Company.

[42] —[1967]. **Mathematical Logic.** Wiley.

[43] —[1981]. *Origins of recursive function theory.* **Annals of History of Computing.** vol. 3. pp. 52-65.

[44] Langford, C. H. [1928]. *Concerning logical principles.* **Bulletin of the American Mathematical Society.** vol. 28. pp. 16-40.

[45] Löwenheim, L. [1915]. *On the possibilities in the calculus of relatives.* en [25]].

[46] Manzano, M. [1996]. **Extensions of First Order Logic.** Cambridge: Cambridge University Press.

[47] Odifreddi, P. [1989]. **Classical Recursion Theory.** Amsterdam: North Holland Publishing Company.

[48] Post, E. L. [1921]. *Introduction to a general theory of elementary propositions. (i*n [25]).

[49] —[1927]. *Finite combinatory processes.* **The Journal of Symbolic Logic**, vol. 2, pp. 103-105.

[50] Shoenfield, J. R. [1967]. **Mathematical Logic.** Reading: Addison-Wesley Publishing Company.

[51] —[1995]. *The mathematical work of S. C. Kleene.* **Bulletin of Symbolic Logic**. vol. 1. Núm 1. pp 9- 43.

[52] Smullyan, R. [1993]. **Recursion Theory for Metamathematics.** Oxford Logic Guides 22. Oxford: Oxford University Press.

[53] Russell, B. [1908]. *Mathematical logic as based on the theory of types.* (in [25])

[54] Russell, B. y Whitehead, A. [1910-13]. **Principia Mathematica.** vol. 1-3. Cambridge: Cambridge University Press.

[55] Tarski, A., Mostowski, A. y Robinson, R. M. [1953]. **Undecidable Theories.** Amsterdam: North Holland Publishing Company.

[56] Turing, A. [1936]. *On computable numbers with an application to the ensheidungsproblem.* **Proceedings of the London Mathematical Society**, vol. 42, 1936-37. pp. 230-265. also vol. 43, 1937, pp. 544-546.

[57] —[1937]. *Computability and $\lambda-$definability.* **The Journal of Symbolic Logic**, vol. 2, pp. 153-163.