

It's Computable – thanks to Alonzo Church



Alonzo Church (1903-1995) @ University of Berkeley

You know, the fact that you can read your email on a cell phone as well as on your desktop computer or almost any other computer connected to the internet, in principle is possible thanks to mathematician **Alonzo Church**, who gave the proof (together with Alan Turing) that everything that is computable on the simple model of a Turing Machine, also is computable with any other ‘computer model’.[8]

Academic Career

Alonzo Church was born on June 14, 1903 in Washington, D.C., where his father, Samuel Robbins Church, was a judge at the Municipal Court for the District of Columbia. The family later moved to Virginia after his father lost the job due to poor eyesight. With the help of his uncle, also known as Alonzo Church, the son attended the private Ridgefield School for Boys in Ridgefield, Connecticut. After graduating from Ridgefield in 1920, Church studied at Princeton University and graduated with a doctorate. His doctoral work was supervised by Oswald Veblen,[12] and he was awarded his doctorate in 1927 for his dissertation entitled *Alternatives to Zermelo’s Assumption*. After stays at the University of Chicago, the Georg August University of Göttingen and the University of Amsterdam he became Princeton Professor of Mathematics in 1929.

The Lambda Calculus

He became known to his mathematical-logical colleagues for his development of the Lambda Calculus, to which he wrote in a report published in 1936 (Church-Rosser’s theorem) in which he demonstrated that there are undecidable problems (i.e. the answer to a question cannot be calculated mathematically). The lambda calculus is a formal language for the investigation of functions. Alonzo Church used the lambda calculus in 1936 both to give a negative answer to the decision problem and to find a foundation for a logical system, as was the case with the *Principia Mathematica* of Bertrand Russell and Alfred North Whitehead. By means of the untyped Lambda calculus one can clearly define what a calculable function is. The question whether two lambda expressions (see below) are equivalent cannot generally be decided algorithmically. In its typified form, the calculus can be used to represent higher level logic. The Lambda Calculus has significantly influenced the development of functional programming languages, research on type systems of programming languages in general, and modern sub-disciplines in logic such as type theory.

The Entscheidungsproblem

In mathematics and computer science, the ‘Entscheidungsproblem’ is one of the challenges posed by mathematician David Hilbert in 1928.[9] The Entscheidungsproblem asks for an algorithm that takes as input a statement of a first-order logic and answers “Yes” or “No” according to whether the statement is universally valid, i.e., valid in every structure satisfying the underlying axioms. Actually, the origin of the Entscheidungsproblem goes back to Gottfried Wilhelm Leibniz, who in the 17th century, after having constructed a successful mechanical calculating machine, dreamt of building a machine that could manipulate symbols in order to determine the truth values of mathematical statements.[10] Leibniz realized that the first step would have to be a clean formal language, and much of his subsequent work was directed towards that goal.

The Church-Turing Thesis

By the completeness theorem of first-order logic, a statement is universally valid if and only if it can be deduced from the axioms, so the Entscheidungsproblem can also be viewed as asking for an algorithm to decide whether a given statement is provable from the axioms using the rules of logic. In 1936 and 1937, Alonzo Church and his student Alan Turing, respectively, published independent papers showing that a general solution to the Entscheidungsproblem is impossible. To achieve this, Alonzo Church applied the concept of “effective calculability” based on his Lambda calculus, while Alan Turing based his proof on his concept of Turing machines. Church then also found that the Lambda calculus and the Turing machine were equal in expressiveness, and were able to give some more equivalent mechanisms for calculating functions. A thesis for the intuitive calculability concept derived from this is known as the Church-Turing thesis. The lambda calculus also influenced the design of the LISP programming language and functional programming languages in general.

Relating to Gödel’ Incompleteness Theorem

It was recognized immediately by Turing that these two concepts are equivalent models of computation. Both authors were heavily influenced by Kurt Gödel’s earlier work on his incompleteness theorem, especially by the method of assigning numbers (also-called Gödel numbering) to logical formulas in order to reduce logic to arithmetic. Church’s Theorem, showing the undecidability of first order logic, appeared in *A note on the Entscheidungsproblem* published in the first issue of the *Journal of Symbolic Logic*. This, of course, is in contrast with the propositional calculus which has a decision procedure based on truth tables. Church’s Theorem extends the incompleteness proof given of Gödel in 1931.[11]

Further Contributions

Church was a founder of the *Journal of Symbolic Logic* in 1936 and was an editor of the reviews section from its beginning until 1979. Church wrote the classic book *Introduction to Mathematical Logic* in 1956. This was a revised and very much enlarged edition of *Introduction to mathematical logic* which Church published twelve years earlier in 1944. In 1960 Church was elected to the American Academy of Arts and Sciences, in 1978 to the National Academy of Sciences. In 1962 he gave a plenary lecture at the International Congress of Mathematicians in Stockholm (*Logic, Arithmetic and Automata*).

Alonzo Church died on August 11, 1995, aged 92.

References and further Reading:

- [1] [Church’s Theorem](#) in Wikipedia
- [2] Alonzo Church, “[A note on the Entscheidungsproblem](#)“, *Journal of Symbolic Logic*, 1 (1936), pp 40–41.
- [3] Alan Turing, “[On computable numbers, with an application to the Entscheidungsproblem](#)“, *Proceedings of the London Mathematical Society*, Series 2, 42 (1937), pp 230–265.
- [4] O’Connor, John J.; Robertson, Edmund F., “[Alonzo Church](#)“, *MacTutor History of Mathematics archive*, University of St Andrews.
- [5] [Alonzo Church](#) at the Mathematics Genealogy Project

- [6] [Alonzo Church at zbMATH](#)
- [7] [Alonzo Church at Wikidata](#)
- [8] [Churchill's Best Horse in the Barn – Alan Turing, Codebreaker and AI Pioneer](#), SciHi Blog
- [9] [David Hilbert's 23 Problems](#), SciHi Blog
- [10] [Let Us Calculate – the Last Universal Academic Gottfried Wilhelm Leibniz](#), SciHi Blog
- [11] [Kurt Gödel Shaking the Very Foundations of Mathematics](#), SciHi Blog
- [12] [Oswald Veblen and Foundations of modern Topology](#), SciHi Blog
- [13] Andrew Appel: [Turing, Gödel, and Church at Princeton in the 1930s](#), princetonacademics @ youtube
- [14] [Timeline of American Logicians](#), via DBpedia and Wikidata