17
History of Computability

17.1 Hilbert's Programs

Around 1880, Georg Cantor, a German mathematician, invented naive set theory. A small fraction of this is sometimes taught to elementary school children. It was soon discovered that this naive set theory was inconsistent because it allowed unbounded set formation, such as the set of all sets. David Hilbert, the world's foremost mathematician from 1900 to 1930, defended Cantor's set theory but suggested a formal axiomatic approach to eliminate the inconsistencies. He proposed two programs. *First,* Hilbert wanted an axiomatization for mathematics, beginning with arithmetic, and a finitary consistency proof of that system. *Second*, Hilbert suggested that the statements about mathematics be regarded as formal sequences of symbols, and his famous *Entscheidungsproblem* (decision problem) was to find an algorithm to decide whether a statement was valid or not. Hilbert characterized this as the fundamental problem of mathematical logic.

Hilbert retired and gave a special address in 1930 in Königsberg, the city of his birth. Hilbert spoke on the importance of mathematics in science and the importance of logic in mathematics. He asserted that there are no unsolvable problems and stressed, "We must know. We will know." At a mathematical conference preceding Hilbert's address, a quiet, obscure young man, Kurt Gödel, only a year beyond his Ph.D. in 1931, refuted Hilbert's consistency program with his famous incompleteness theorem and changed forever the foundations of mathematics. Gödel soon joined other

R.I. Soare, *Turing Computability*, Theory and Applications of Computability,
DOI 10.1007/978-3-642-31933-4_17

leading figures, Albert Einstein and John von Neumann, at the Institute for Advanced Study in Princeton.

17.2 Gödel, Church, and Recursive Functions

The refutation of Hilbert's first program on consistency gave hope for refuting his second program on the *Entscheidungsproblem*. However, this was no ordinary problem in number theory or analysis. To prove the unsolvability of a certain problem, such as Hilbert's famous Tenth Problem on Diophantine equations of 1900, one must: (1) find a precise mathematical definition for the intuitive idea of algorithm; (2) demonstrate beyond doubt that every algorithm has been captured; (3) prove that no algorithm on the list can be the solution of the Diophantine equation problem.

Work began independently at Princeton and Cambridge. Alonzo Church completed an A.B. degree at Princeton in 1924 and his Ph.D. degree there under Oswald Veblen in 1927. Church joined the Department of Mathematics at Princeton from 1929 until his retirement in 1967, when he moved to UCLA. Church worked from 1931 through 1934 with his graduate student, Stephen Cole Kleene, on the formal system of λ-definable functions. They had such success that in 1934 Church proposed privately to Gödel that a function is effectively calculable (intuitively computable) if and only if it is λ-definable. Gödel rejected this first version of Church's Thesis. In addition, Kleene reported "chilly receptions from audiences around 1933–35 to disquisitions on λ-definability."

However, in the spring of 1934 Gödel lectured on recursive functions. By 1935 Church and Kleene had moved enthusiastically to the formalism of Gödel's recursive functions as a vehicle to capture the intuitive idea of effectively calculable. In 1931 Gödel had used the *primitive* recursive functions, those where one computes a value $f(n)$ by using previously computed values $f(m)$, for $m < n$, such as the factorial function $f(0) = 1$ and $f(n + 1) = (n + 1) \cdot f(n)$.

In his 1934 lectures at Princeton, Gödel extended this to the (Herbrand-Gödel) (*general*) recursive functions. Church eagerly embraced them and formulated his famous *Church's Thesis* in [Church 1935] and [Church 1936] that the effectively calculable functions coincide with the recursive functions. Again, Gödel failed to accept this thesis even though he was the author of the recursive functions. Gödel noted that recursive functions are clearly effectively calculable, but the converse "cannot be proved, since the notion of finite computation is not defined, but it serves as a heuristic principle."