The Role of Mathematical Logic in the Birth of Computer Science

Cantors set theory: infinite objects in proofs. Mathematicians worry and argue. Paradoxes.

The formalization of mathematical proofs from Leibniz to Boole to Frege. Frege formalizes the predicate calculus and a form of set theory. Russells Paradox.

Hilberts interest in the foundations of mathematics. His Second Problem.

Russell and Whiteheads "Principia Mathematica", a ramified theory of types. Zermelo-Fraenkel set theory.

Hilberts Program. Relative consistency results. Gödels Completeness Theorem. The Decision Problem (Entscheidungsproblem).

Schönfinkel and later Curry invent a (variable-free) functional formalism, combinatory logic. Church invents the (variable-full) lambda calculus as a skeleton for an alternative to the Principia system and to ZF set theory.

Gödel shatters Hilberts Program with his Incompleteness Theorems. Kleene encodes arithmetic in the lambda calculus. He and Rosser show that the Church and Curry proposals for logics based on functional formalisms are inconsistent.

Church proposes lambda definability as a formal definition of computability/decidability (Churchs Thesis). He shows that there exist undecidable problems in arithmetic and that the Entscheidungsproblem is undecidable.

Kleene shows that lambda definability is equivalent to Gödels proposal for computability based on an idea of Herbrand (definability by systems of arithmetic equations).

Independently of Church, Turing shows that the Entscheidungsproblem is undecidable. His proposal for computability relies, of course, on what we call now Turing Machines and, critically, on the Universal Turing Machine. After he becomes aware of the work by Church and Kleene he shows that his notion of computability is equivalent to lambda definability. Gödel has written that it was Turing's work that finally gave a convincing definition of computability.

Inspired by his work on lambda calculus Kleene develops recursion theory, out of which complexity theory emerges later.

Church develops the simply typed lambda calculus as the foundation of a simple theory of types. This leads eventually to modern proof assistants based on type theory.

Inspired by the Universal Turing Machine, von Neumann and, apparently, his collaborators in the Eniac project propose the computer architecture that bears von Neumann's name.