IR 00053
June 12, 1956

COMPUTER DECISIONS IN DEDUCTIVE LOGIC

The work proposed, of developing a programming technique for handling axiomatic systems in algebraic form, is directed towards extending the ability of existing digital computers. We shall develop a program, based on the natural deductive systems of Quine and Fitch rather than a decision procedure, which will enable the 704 computer to perform logical derivations in a style that can be duplicated only by a trained logician. Concepts of this program may then be extended to quantification theory, mathematical induction, and general mathematical proofs.

Trenchard More

The work proposed, of developing a programming technique for handling axiomatic systems in algebraic form, is directed towards extending the ability of existing digital computers. The majority of present programs, making repeated use of a few arithmetic and logical operations, can direct a machine to solve complicated problems related to arithmetic, calculus and economics. Programs of this nature depend on the arithmetic significance of the data, in that the data are manipulated as numbers or arithmetic magnitudes. Arithmetic operations and relations are used implicitly in the program to manipulate the numbers denoted by the input signals. The remaining programs, using the same simple operations, can direct a machine to solve problems related to language translation, inventory control and business decisions.

Programs of this second nature depend on the symbolic significance of the data, in that the data are manipulated as labels. A more general class of operations and relations are used in the program to manipulate the symbols depicted by the input signals.

Programs of these present two classes use implicitly operations and relations to transform input numbers and symbols into output numbers and symbols. We are interested in the problem of using implicitly rules of inference in the program to algebraically transform input operations and relations on symbols into output operations and relations on symbols.

For example, a theorem of symbolic logic might be transformed into the logical derivation or proof of that theorem by a program operating on axioms, rules of inference, and previously learned theorems. The general methods of a program which succeeds in deriving given equations from a few axioms by rules of inference, can be extended to program problems in large areas of mathematics, such as lattice theory and topology, which are not founded on arithmetic. Arithmetic problems may also be solved by the more powerful methods of algebra and mathematical induction.

Although the proposed work concerns a deductive system of unquantified statements based only on Boolean algebra, we do not intend to follow a decision procedure such as truth table analysis. A decision procedure for an axiomatic system is a demonstrated method of deciding definitely for each particular statement expressed in the language of the system, whether that statement is valid. A decision procedure need not belong to the language of the system, whereas the steps of a derivation are written in the same language as the statement to be derived. The reason for programming an uncertain deductive routine that depends on guesses and theorems learned by the machine, rather than a simpler and more direct decision procedure, is that more complicated and useful mathematical systems admit no decision procedure whatever.

In the quantified predicate calculus of multivalued functions, a decision procedure does not exist, and proofs are left to the ingenuity of the mathematician or machine. The proposed program can be extended directly to quantification theory and then to systems depending on mathematical induction, whereas a program based on a decision procedure would have to be revised completely.

Although not admitting a decision procedure, general quantification theory is complete, in that every valid statement can be derived, even though one might lack the ingenuity to find the derivation, and consistent in that contradictions cannot be derived. Hilbert wanted to show that all the axiomatic systems containing classical mathematics had these same desirable properties. This hope of consistency and completeness in the formalization of classical mathematics was shattered in 1931 by Godel's two theorems "on formally undecidable propositions of Principia Mathematica and related systems". The first of these theorems as extended by Rosser states that, if a formal axiomatic system A (including some of Peano's axioms of arithmetic) is consistent, then the system A is incomplete; and the incompleteness is shown by exhibiting a formula of A which can neither be proved nor disproved by arguments within A.

The second theorem states that, if the system A is consistent, then there is no way of reasoning within A that will prove the system A to be consistent.

Comparatively simple axiomatic systems, such as arithmetic, whose elements are easy to understand, contain statements that can be proved only by ingenuity if proof is even possible. It is hoped that the methods begun in the proposed research will extend eventually to programs that will help mathematicians prove difficult theorems.

There is a major result of meta-mathematics that is particularly relevant to computing machines. Godel's and Kleene's concept of general recursiveness, Church's concept of lambda--definability, and Turing's concept of computability were shown to be entirely equivalent by those authors in 1936 and 1937. Recursive functions are defined by an iterative procedure that is analogous to the methods of mathematical induction. The concept of recursion gives logicians a genetic way of developing definitions, formulas and statements.

It can be shown that every general recursive function has a decision procedure. One may ask why, if computers are confined to recursive operations, we do not use a decision procedure. Given a

program, a computer without random elements is bound to decision procedure methods. We intend to have the computer learn pertinent theorems from its operator. In this way, the capricious and selective elements so necessary to intelligence are supplied from a human source.

As a specific example, we shall develop a program of instructions which will enable the 704 computer to derive certain theorems of logic. After being given a statement, such as

John is ill and two equals two infers that John is ill

in symbolic form

$$\left[(\text{John is ill}) \wedge (2 = 2)\right] \vdash (\text{John is ill})$$

the machine will either derive from a few basic assumptions the steps of a deduction that prove or disprove the statements, or else give up. Evaluated in terms of its inputs and outputs, the machine will perform logical derivations in a style that can be duplicated only by a trained logician.

The report of this work should have at least four parts.

1. The mathematical development of the deductive system used as a model for the program.

2. A discussion of the general principles used to transform the deductive system to a program.

3. An analysis and description of the program itself.

4. An evaluation of the results obtained from machine operation.

Since this work is also being done for a master's thesis, we plan to work mostly on the development of a program during working hours, and the rest during evenings. In this way, we hope to complete the program and the thesis by the end of the summer.