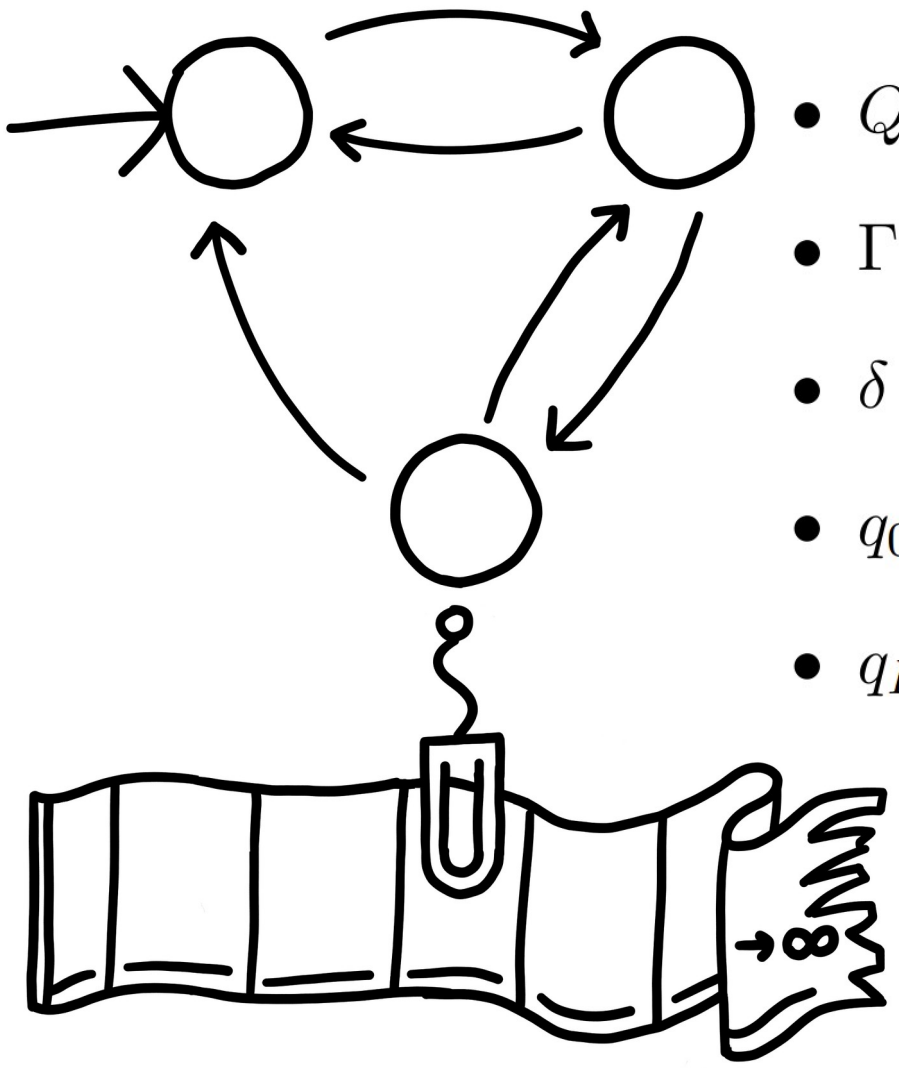
The Church-Turing Thesis

Abrahim Ladha



A Turing Machine is a tuple $(Q, \Gamma, \delta, q_0, q_H)$

- $Q = \{q_0, ..., q_k\}$ is a finite set of states
- $\Gamma = \{0, 1, \sqcup\}$ finite tape alphabet
- $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$
- $q_0 \in Q$ designated start state
- $q_H \in Q$ is the instruction to stop

LEMMA 1. *If $\theta_A(i_0)$ is defined, then $\theta_A(i_0) \in \{i_0\}_A$, $\theta_A(i_0) > 2i_0$.*

PROOF. If $\theta_A(i_0)$ is defined, then there is some value of t such that $T^A(i_0, K(t), L(t))$, $\theta(i_0) = K(t)$, and $\theta_A(i_0) > 2i_0$. Hence,

$$\bigvee_y T^A(i_0, \theta_A(i_0), y); \text{ that is, } \theta_A(i_0) \in \{i_0\}_A.$$

LEMMA 2. *If $\{i\}_A$ is infinite, then $\{i\}_A \cap S^A \neq \phi$.*

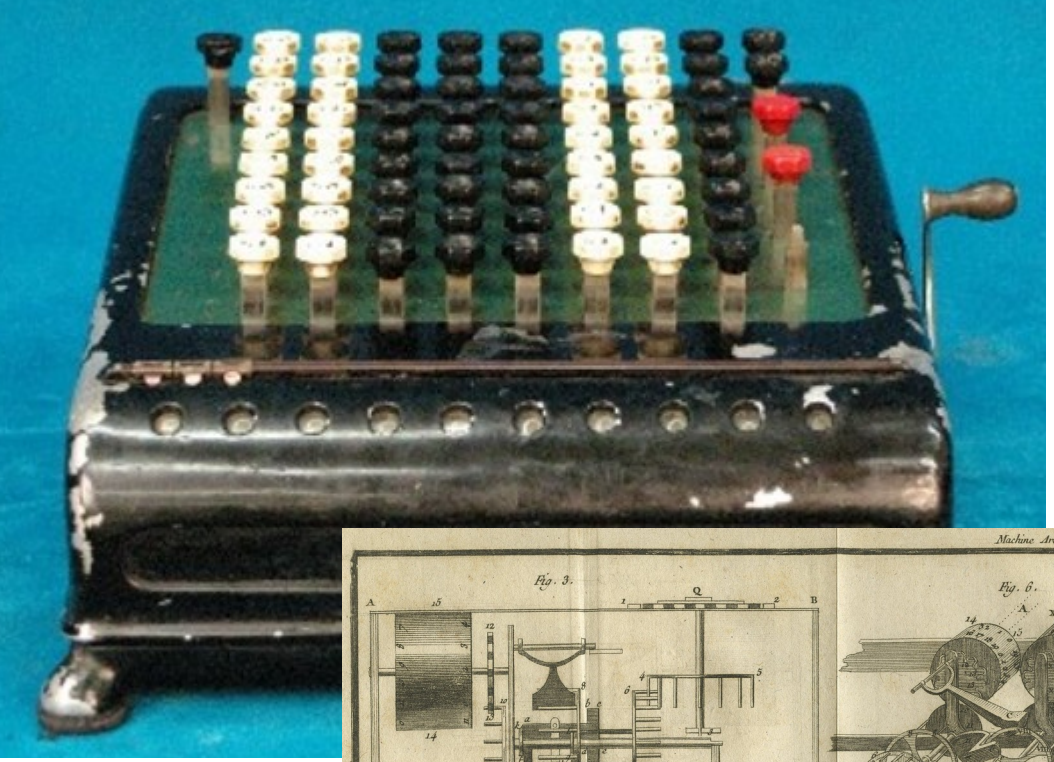
PROOF. If $\{i_0\}_A$ is infinite, there is a number m_0 such that $m_0 \in \{i_0\}_A$ and $m_0 > 2i_0$. Hence, $\bigvee_y T^A(i_0, m_0, y)$. Let y_0 be the least such y. (Actually, by the definition of the T-predicates there is at most one such y.) Then $T^A(i_0, m_0, y_0)$. Let $t_0 = J(m_0, y_0)$. Then

$$m_0 = K(J(m_0, y_0)) = K(t_0); \; y_0 = L(J(m_0, y_0)) = L(t_0).$$

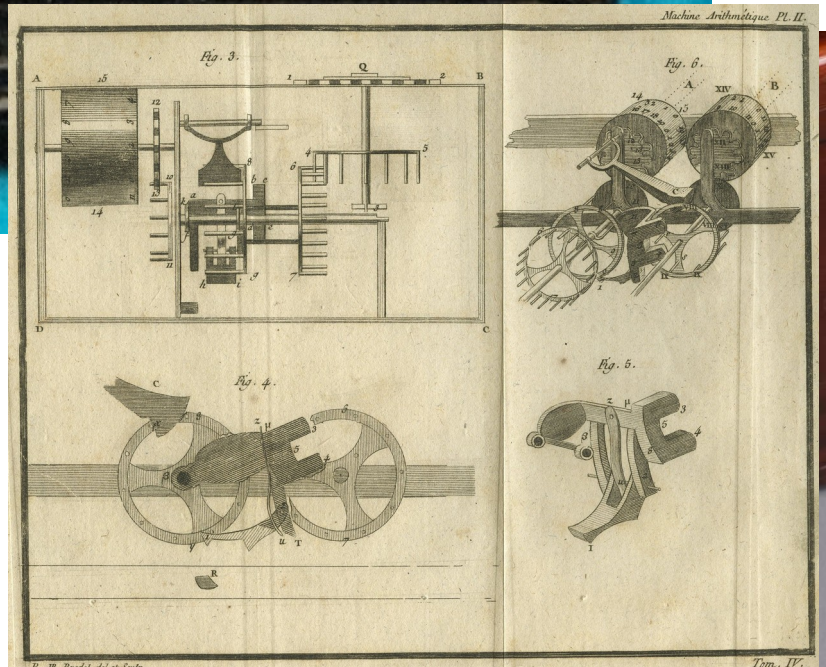
Hence, $T^A(i_0, K(t_0), L(t_0))$. Since $K(t_0) = m_0 > 2i_0$, $\theta_A(i_0)$ is defined. Hence, by Lemma 1, $\theta_A(i_0) \in \{i_0\}_A$. But, $\theta_A(i_0) \in S^A$. Hence, $\{i_0\}_A \cap S^A \neq \phi$.

For a set of recursion equations for F^* consists of the recursion equations for F together with the equations,

$$i_2(1, 2) = 2,$$
$$i_2(S(x), 2) = 1,$$
$$i_2(x, 1) = 3,$$
$$i_2(x, S(S(y))) = 3,$$
$$j_2(1, y) = y,$$
$$j_2(S(x), y) = x,$$
$$g_2(x, 1) = i_2(f_2(x, 1), 2),$$
$$g_2(x, S(y)) = i_2(f_2(x, S(y)), g_2(x, y)),$$
$$h_2(S(x), y) = x,$$
$$h_2(g_2(x, y), x) = j_2(g_2(x, y), y),$$
$$f_1(x) = h_2(1, x),$$



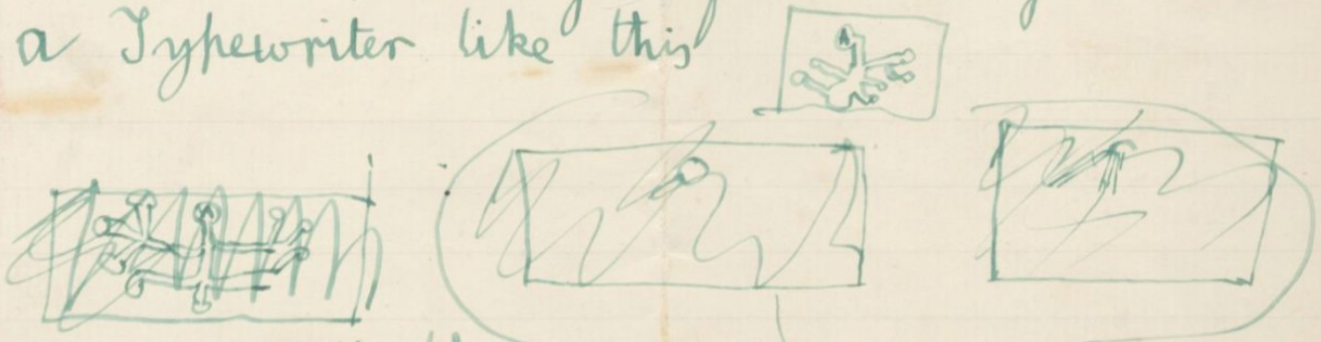

ROYAL


Machine Arithmétique Pl. II.
Fig. 3.
Fig. 6.
Fig. 4.
Fig. 5.
Tom. IV.

[3.a]

[No 3a]

It is luck I don't have to show up my letters* or Mr Darlington would conversecate the whole bottle# probably. This week I thought of how I might make a Typewriter like this



you see the the funny little rounds are letters cut out on one side slide along to the round (A) and along an ink pad and stamp down and make the letter, thats not nearly all though

Love from

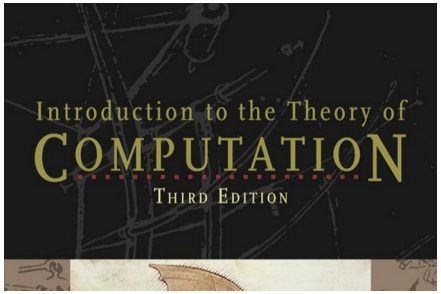
Alan

What is the Church-Turing Thesis?

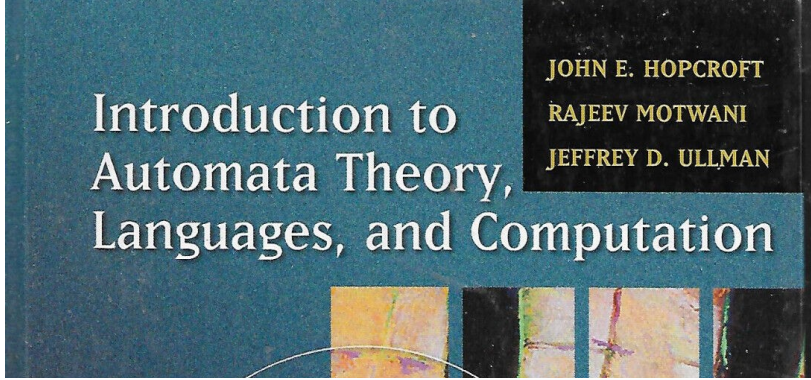
- “Algorithm” is an informal, intuitive notion of a process. A set of instructions to complete some task.
- A Turing Machine is a formal definition of a model of computation
- The Church-Turing Thesis asserts that the Turing machine captures the intuitive definition. That everything computable in the intuitive sense is computable by a Turing machine. But why?

Informally speaking, an *algorithm* is a collection of simple instructions for carrying out some task. Commonplace in everyday life, algorithms sometimes are called *procedures* or *recipes*.

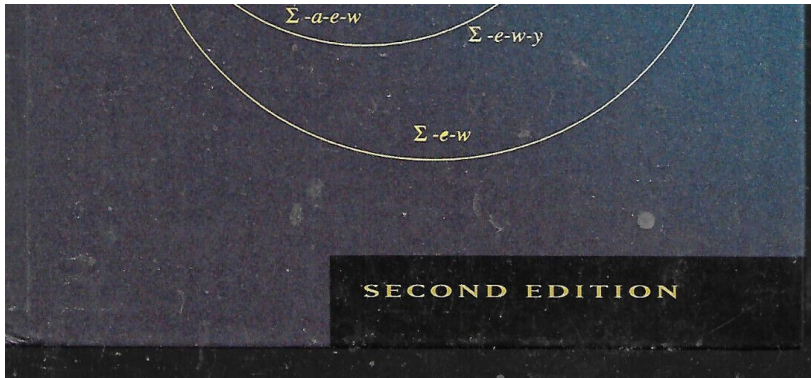
Even though algorithms have had a long history in mathematics, the notion of algorithm itself was not defined precisely until the twentieth century. Before that, mathematicians had an intuitive notion of what algorithms were, and relied upon that notion when using and describing them. But that intuitive notion was insufficient for gaining a deeper understanding of algorithms.



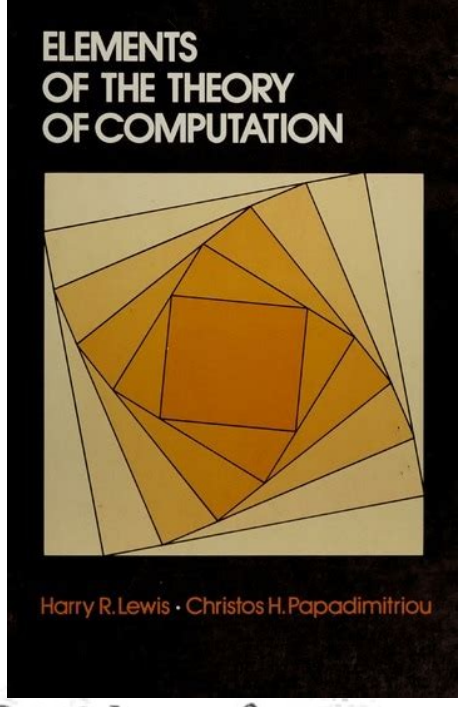
The definition came in the 1936 papers of Alonzo Church and Alan Turing. Church used a notational system called the λ-calculus to define algorithms. Turing did it with his "machines." These two definitions were shown to be equivalent. This connection between the informal notion of algorithm and the precise definition has come to be called the ***Church–Turing thesis***.



Interestingly, all the serious proposals for a model of computation have the same power; that is, they compute the same functions or recognize the same languages. The unprovable assumption that any general way to compute will allow us to compute only the partial-recursive functions (or equivalently, what Turing machines or modern-day computers can compute) is known as *Church's hypothesis* (after the logician A. Church) or the *Church-Turing thesis*.



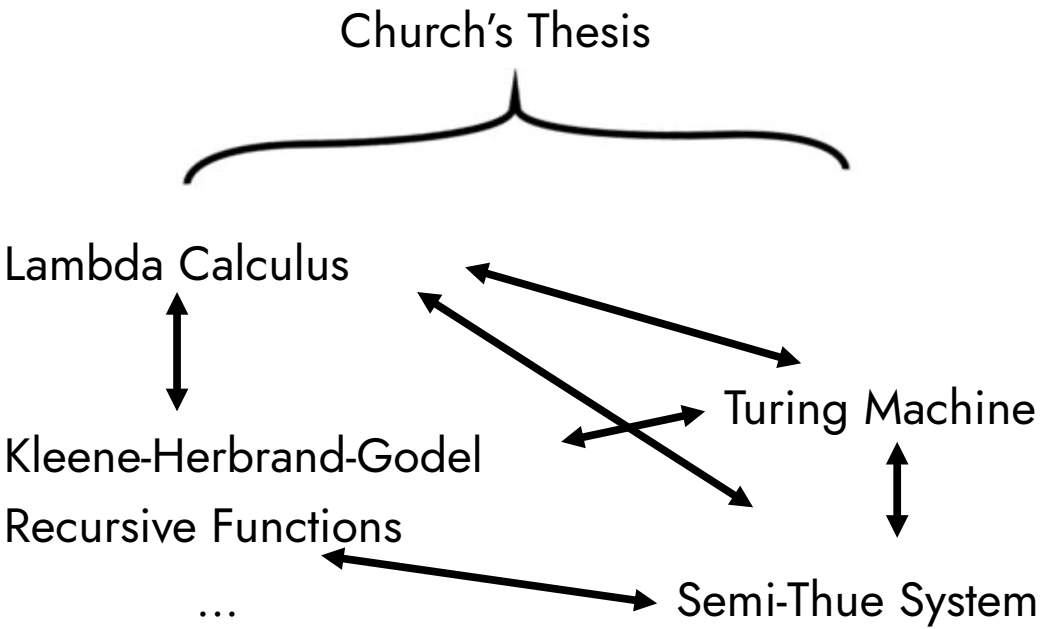
The situation is quite analogous to that met whenever one attempts to replace a vague concept, having a powerful intuitive appeal, with an exact mathematical substitute. (An obvious example is the area under a curve.) In such a case, it is, of course, pointless to demand a mathematical proof of the equivalence of the two concepts; the very vagueness of the intuitive concept precludes this.



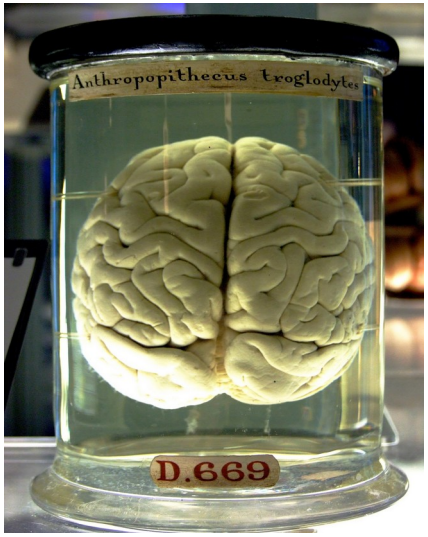
Turing machines can be imitated by grammars,
which can be imitated by μ-recursive functions,
which can be imitated by Turing machines.

The only possible conclusion is that *all these approaches to the idea of computation are equivalent*. This is Church's Thesis, extended now to methods quite different from those of the theory of automata.

considering one aspect of computation by


Church's Thesis
Lambda Calculus
Turing Machine
Kleene-Herbrand-Godel
Recursive Functions
...
Semi-Thue System

=????


Anthropopithecus troglodytes
D.669

Human Mind

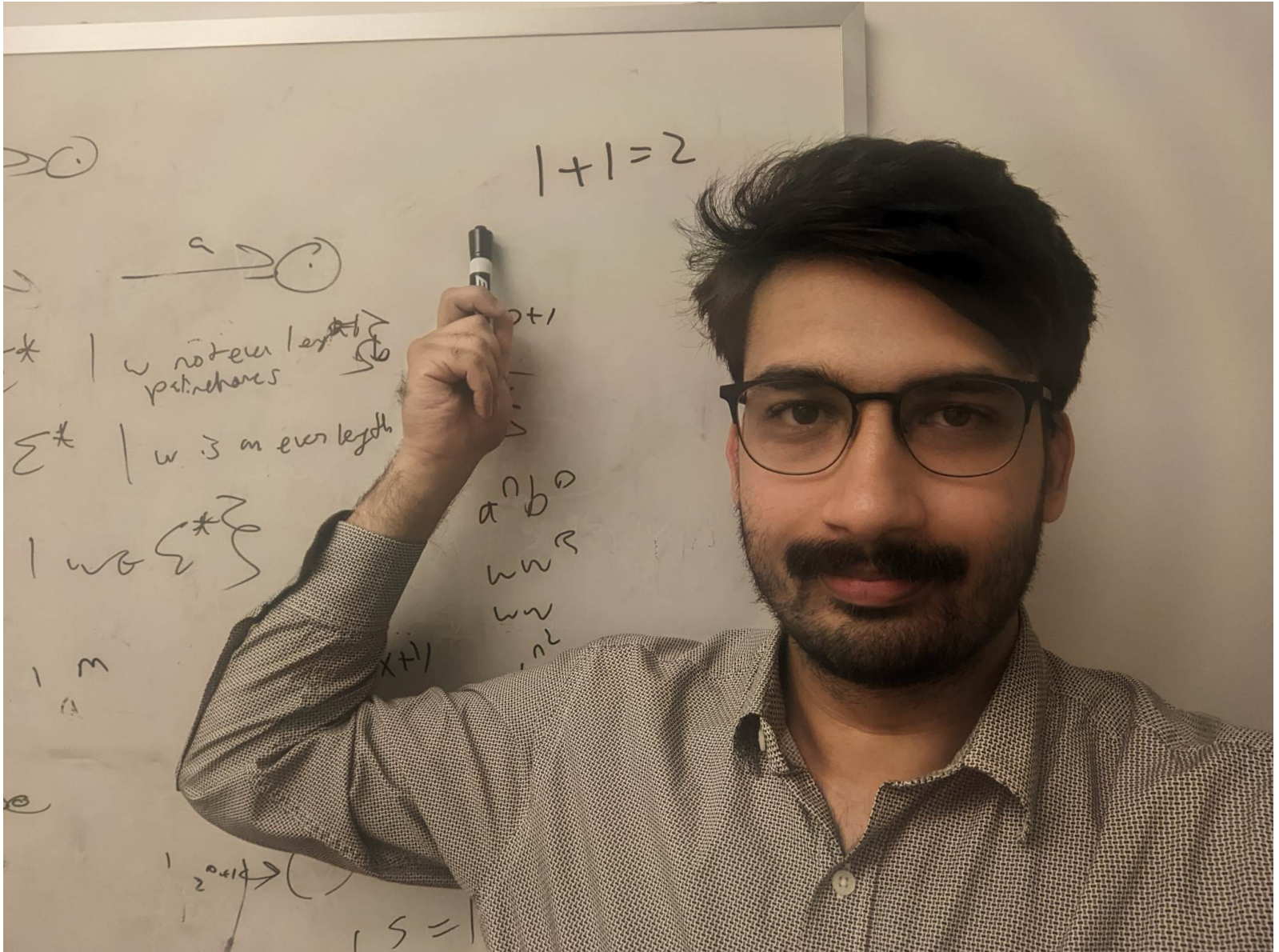
ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM

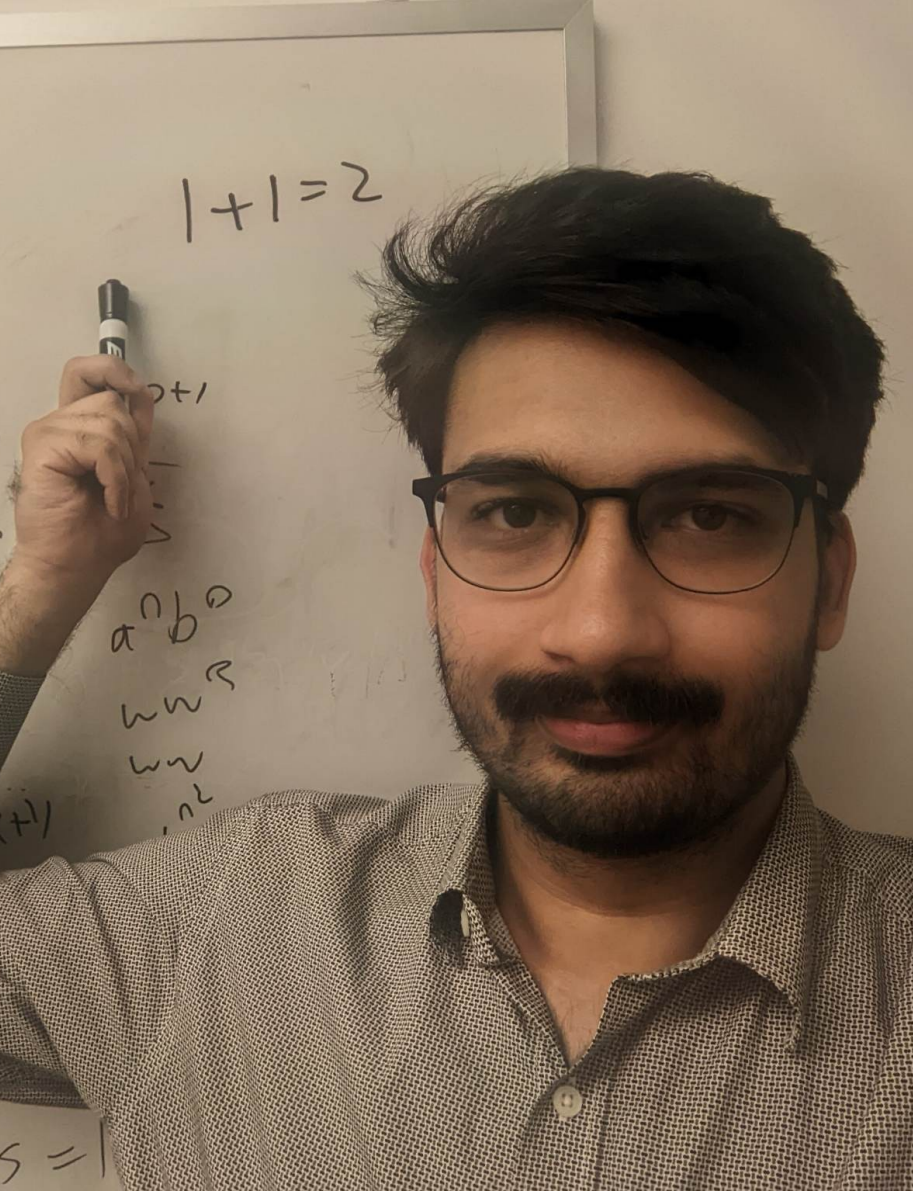
By A. M. TURING.

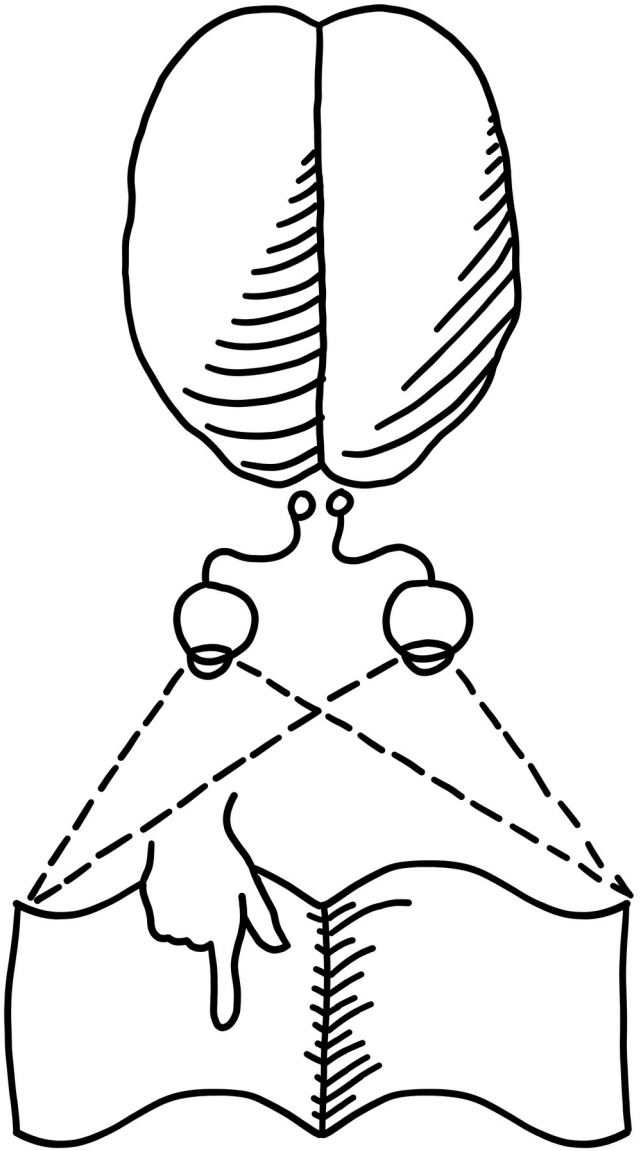
[Received 28 May, 1936.—Read 12 November, 1936.]

The Direct Appeal to Intuition

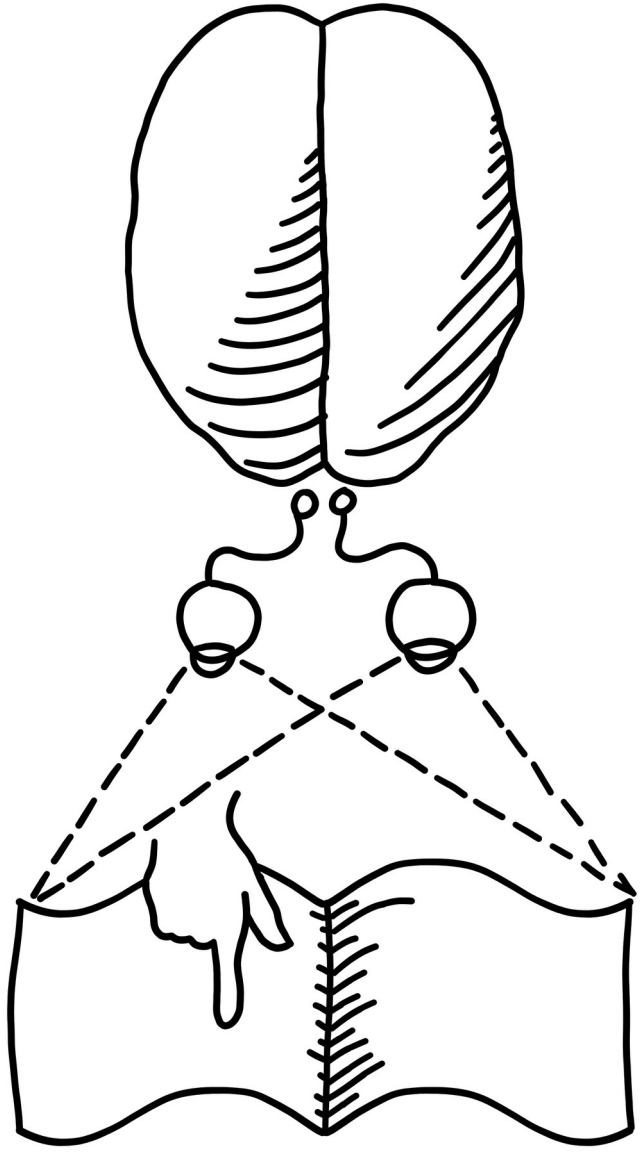
- First we will agree on an example of a computer, an object performing the action of computation
- We will distill the example until it is simplified but still undisagreeable
- We will make small changes, not enough to change the fact it is a computer
- We will argue that the composition of these modifications is still correct and conclude

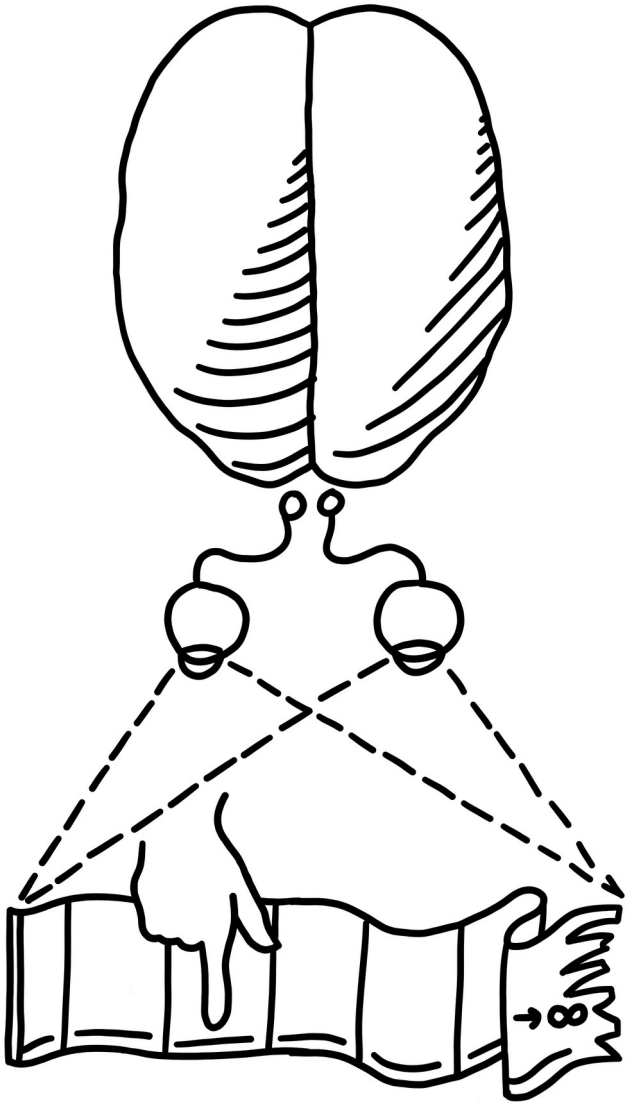

1+1=2


1+1=2



Computing is normally done by writing certain symbols on paper. We may suppose this paper is divided into squares like a child's arithmetic book. In elementary arithmetic the two-dimensional character of the paper is sometimes used. But such a use is always avoidable, and I think that it will be agreed that the two-dimensional character of paper is no essential of computation. I assume then that the computation is carried out on one-dimensional paper, *i.e.* on a tape divided into squares.

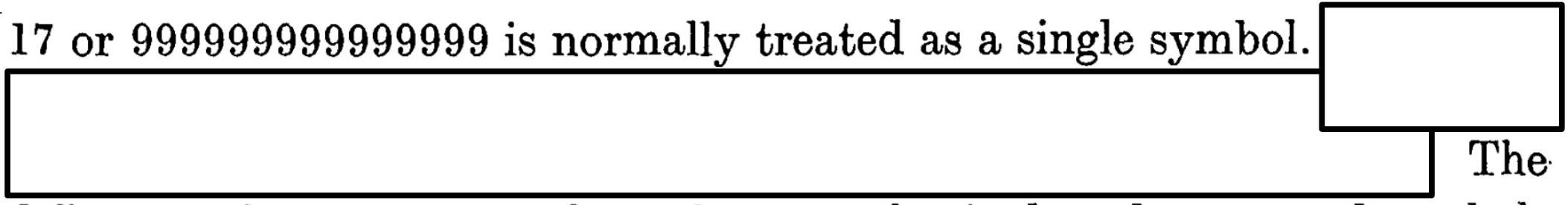




I shall also suppose that the number of symbols which may be printed is finite. If we were to allow an infinity of symbols, then there would be symbols differing to an arbitrarily small extent†. The effect of this restriction of the number of symbols is not very serious. It is always possible to use sequences of symbols in the place of single symbols. Thus an Arabic numeral such as

---

17 or 99999999999999 is normally treated as a single symbol.

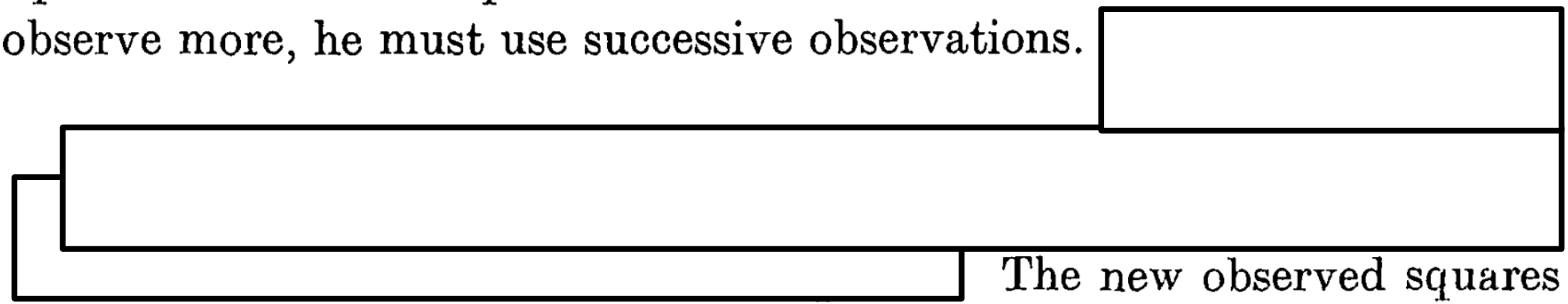


The differences from our point of view between the single and compound symbols is that the compound symbols, if they are too lengthy, cannot be observed at one glance. This is in accordance with experience. We cannot tell at a glance whether 9999999999999 and 99999999999999 are the same.

- Finite work is done in finite time, important for the development of computational complexity
- The amount of symbols necessary for computation must be finite

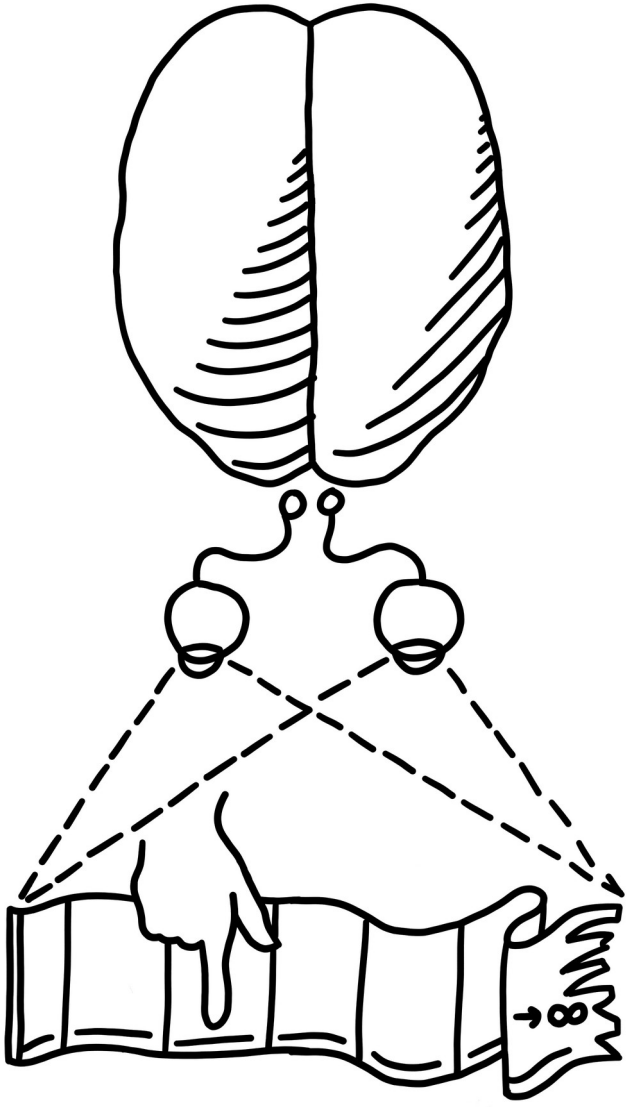
$$|\Sigma| < \infty$$

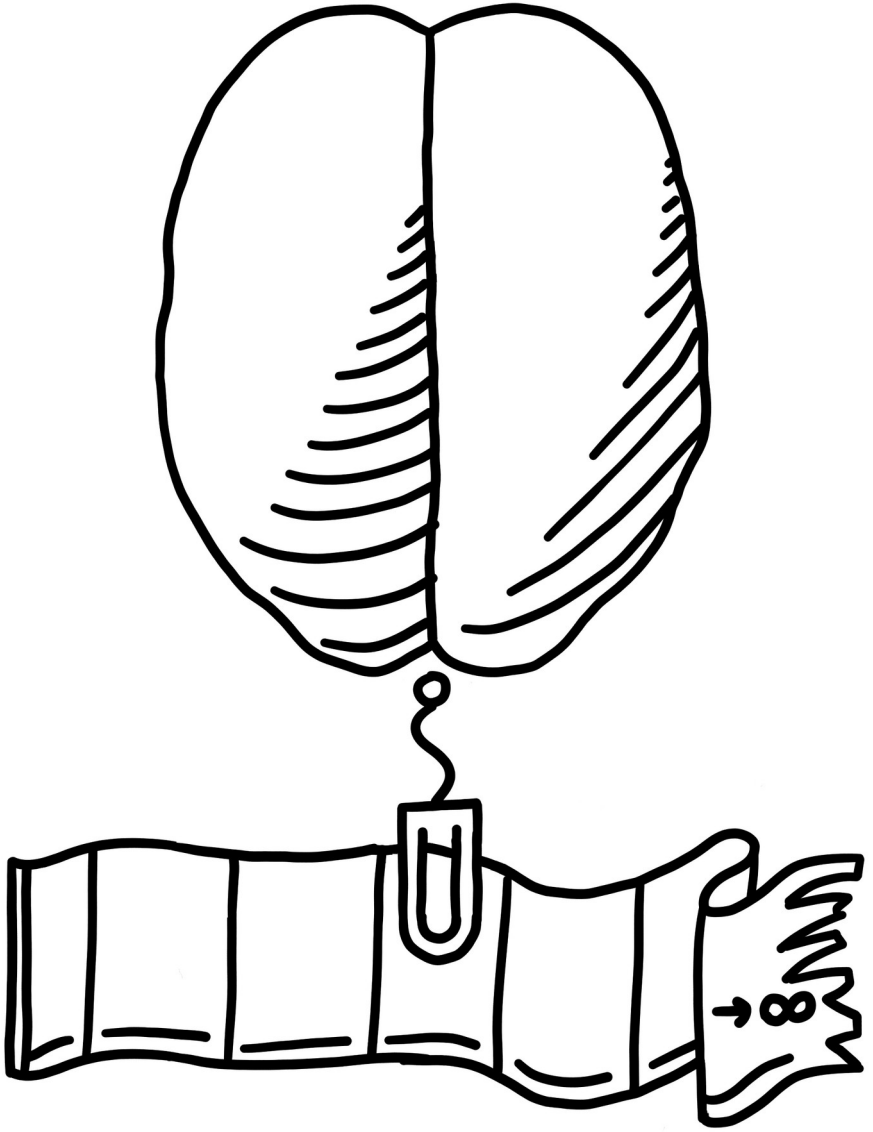
The behaviour of the computer at any moment is determined by the symbols which he is observing, and his "state of mind" at that moment. We may suppose that there is a bound B to the number of symbols or squares which the computer can observe at one moment. If he wishes to observe more, he must use successive observations.



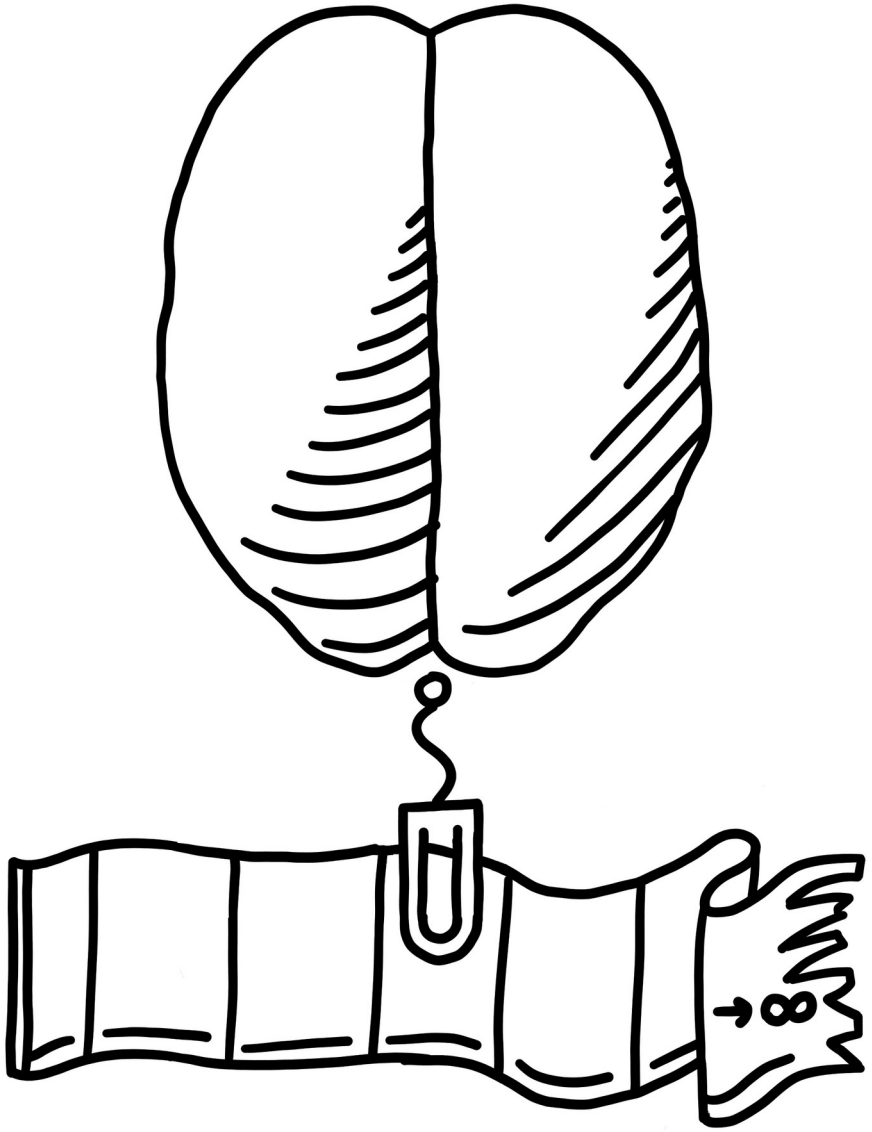
The new observed squares must be immediately recognisable by the computer. I think it is reasonable to suppose that they can only be squares whose distance from the closest of the immediately previously observed squares does not exceed a certain fixed amount. Let us say that each of the new observed squares is within L squares of an immediately previously observed square.

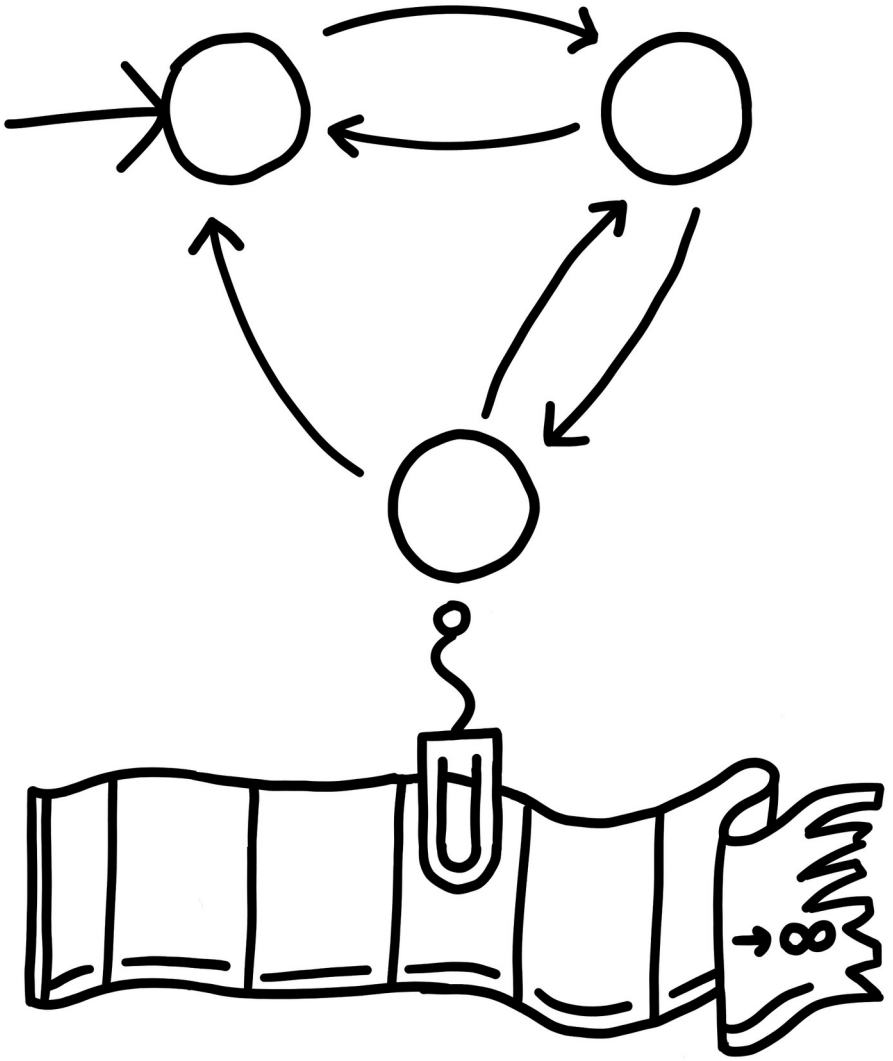
- $L = B = 1$
- δ only need take as input the current state and symbol





We will also suppose that the number of states of mind which need be taken into account is finite. The reasons for this are of the same character as those which restrict the number of symbols. If we admitted an infinity of states of mind, some of them will be "arbitrarily close" and will be confused.





Let us imagine the operations performed by the computer to be split up into "simple operations" which are so elementary that it is not easy to imagine them further divided. Every such operation consists of some change of the physical system consisting of the computer and his tape. We know the state of the system if we know the sequence of symbols on the tape, which of these are observed by the computer (possibly with a special order), and the state of mind of the computer. We may suppose that in a simple operation not more than one symbol is altered. Any other changes can be split up into simple changes of this kind. The situation in regard to the squares whose symbols may be altered in this way is the same as in regard to the observed squares. We may, therefore, without loss of generality, assume that the squares whose symbols are changed are always "observed" squares.

Now if these squares are marked only by single symbols there can be only a finite number of them, and we should not upset our theory by adjoining these marked squares to the observed squares. If, on the other hand, they are marked by a sequence of symbols, we cannot regard the process of recognition as a simple process. This is a fundamental point and should be illustrated. In most mathematical papers the equations and theorems are numbered. Normally the numbers do not go beyond (say) 1000. It is, therefore, possible to recognise a theorem at a glance by its number. But if the paper was very long, we might reach Theorem 157767733443477; then, further on in the paper, we might find "... hence (applying Theorem 157767733443477) we have ...". In order to make sure which was the relevant theorem we should have to compare the two numbers figure by figure, possibly ticking the figures off in pencil to make sure of their not being counted twice. If in spite of this

- Γ need not only contain input symbols, but may contain additional symbols used for computation
- Γ = {a,b,~~a~~,~~b~~,_}
- ~~ab~~aa...#~~ab~~aa...

The simple operations must therefore include:

(a) Changes of the symbol on one of the observed squares.

(b) Changes of one of the squares observed to another square within L squares of one of the previously observed squares.

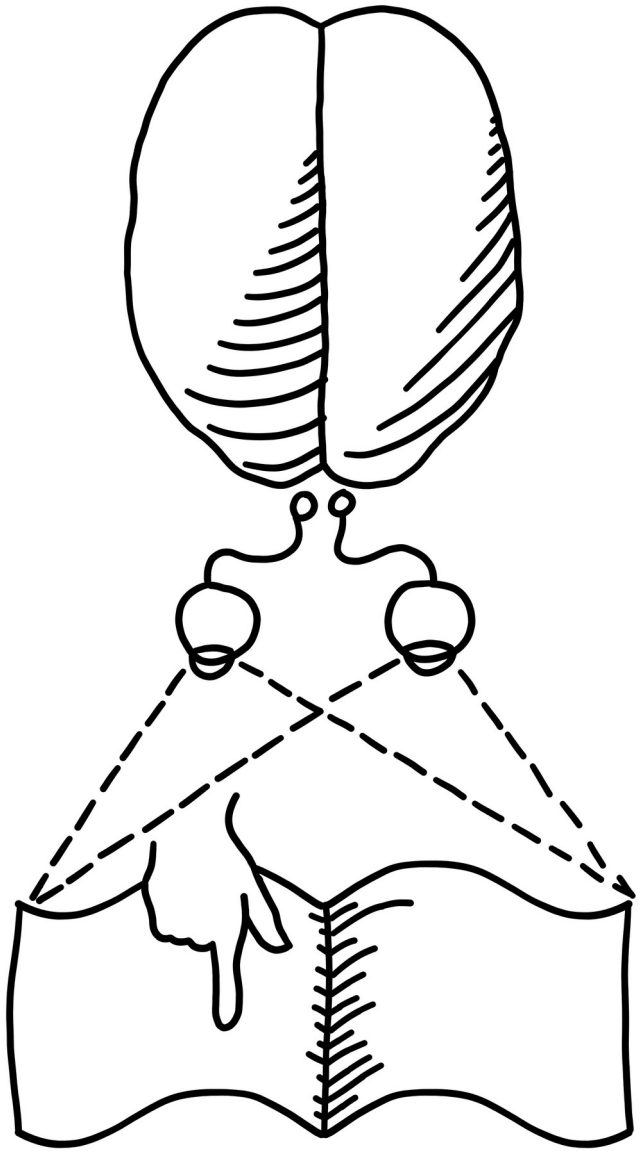
It may be that some of these changes necessarily involve a change of state of mind. The most general single operation must therefore be taken to be one of the following:

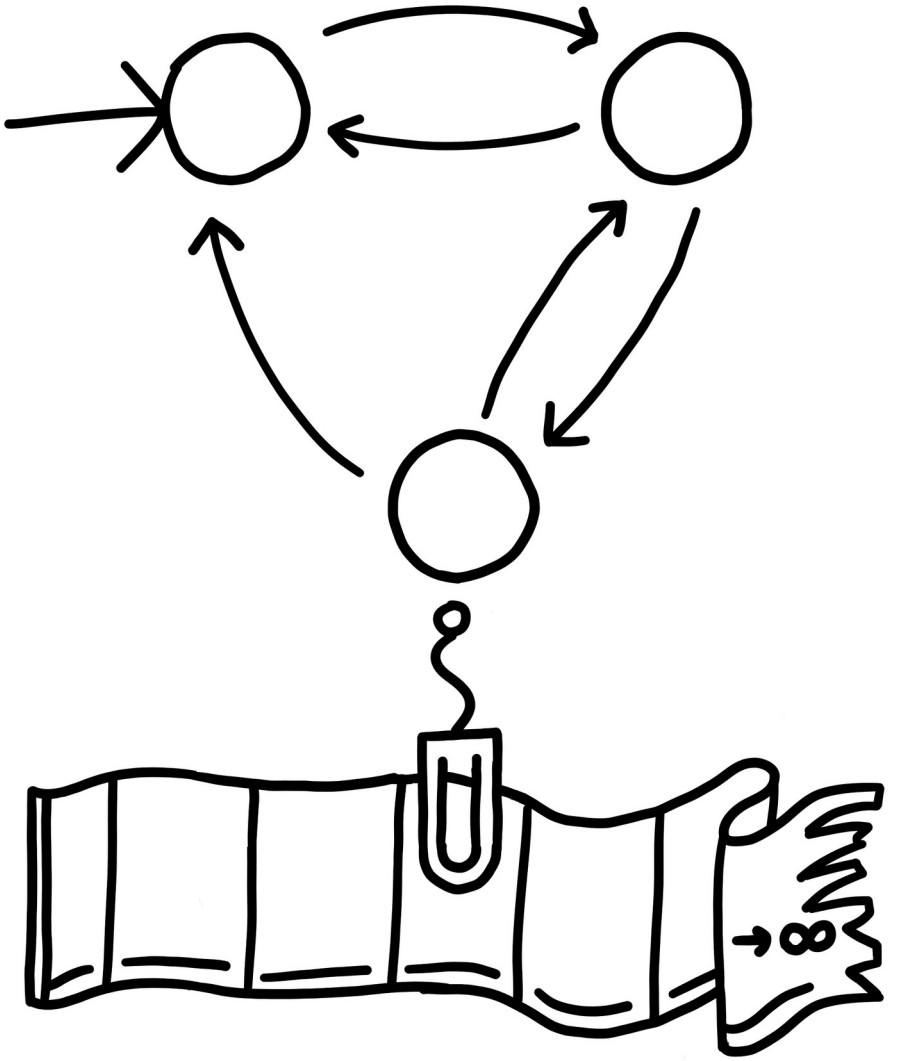
(A) A possible change (a) of symbol together with a possible change of state of mind.

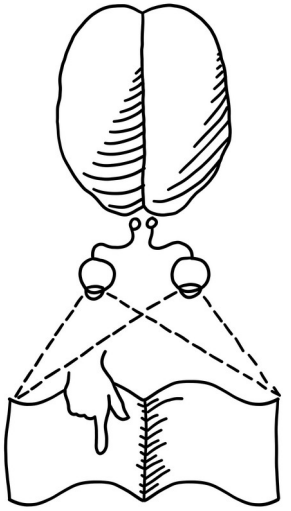
(B) A possible change (b) of observed squares, together with a possible change of state of mind.

- δ only need output the next state, the written symbol, and a left or right move

We may now construct a machine to do the work of this computer. To each state of mind of the computer corresponds an "m-configuration" of the machine. The machine scans B squares corresponding to the B squares observed by the computer. In any move the machine can change a symbol on a scanned square or can change any one of the scanned squares to another square distant not more than L squares from one of the other scanned squares. The move which is done, and the succeeding configuration, are determined by the scanned symbol and the m-configuration. The machines just described do not differ very essentially from computing machines as defined in § 2, and corresponding to any machine of this type a computing machine can be constructed to compute the same sequence, that is to say the sequence computed by the computer.







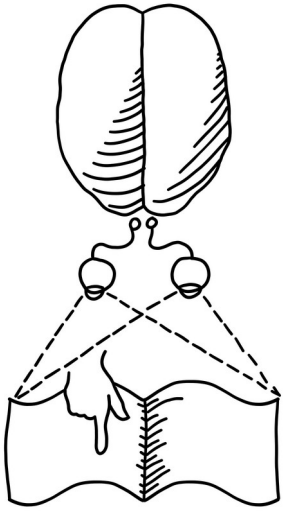
Human Mind

Lambda Calculus

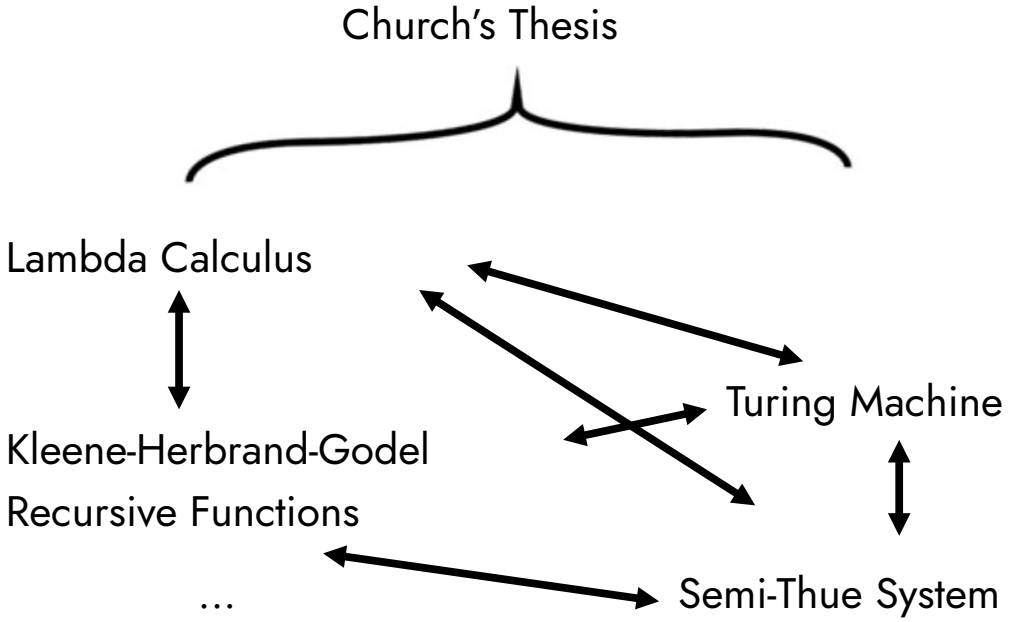
Kleene-Herbrand-Godel
Recursive Functions

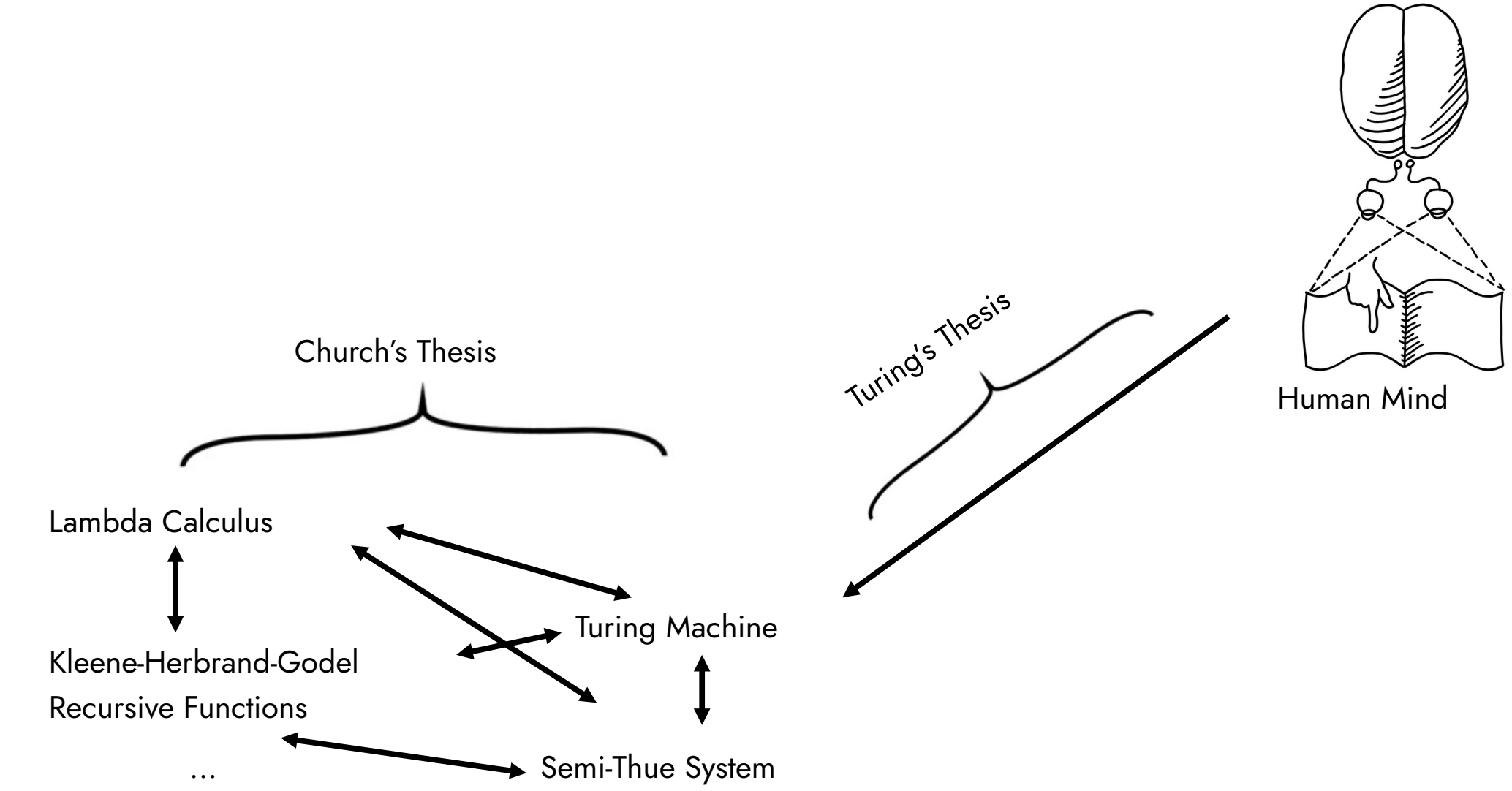
Turing Machine

Semi-Thue System



Human Mind


Church's Thesis
Lambda Calculus
Kleene-Herbrand-Godel
Recursive Functions
...
Turing Machine
Semi-Thue System


Church's Thesis
Turing's Thesis
Human Mind
Lambda Calculus
Turing Machine
Kleene-Herbrand-Godel
Recursive Functions
...
Semi-Thue System

- Church's Thesis: Any reasonable model of computation is equivalent to a Turing machine
- Turing's Thesis: The Turing machine is equivalent in power to the human mind
- Church-Turing Thesis: The definition of algorithm is independent of any specific formalism

Applications

- Well specified pseudocode can always be implemented
- Any results about the formal model (Turing machine) implicate the intuitive notion (algorithm)
- Nonexistence of a Turing machine for some task implies the existence of a problem unsolvable by humans