CM0081 Automata and Formal Languages
The Church-Turing-Kleene Thesis

Andrés Sicard-Ramírez

Universidad EAFIT

Semester 2024-1

A Version of the Thesis

The Church-Turing-Kleene thesis (common version)

A function is effectively calculable if and only if there is a Turing machine which computes it.

Agenda

Goals

(i) Introduction to the thesis.

(ii) To point out that the thesis was not proposed by Church nor Turing but by Kleene.

(iii) To clarify the thesis is not about machines but idealised human computers.

(iv) To clarify the thesis is not about arbitrary functions but number-theoretic functions.

Lambda-Definable Functions and Functions Computable by a Turing Machine

Theorem (imprecise version)

The following sets are coextensive:

(i) the λ-definable functions and

(ii) the functions computable by a Turing machine

Non-Provability of the Thesis

Why the thesis is not a theorem

Informal notion (effectively calculable)

Formal notion (Turing-machine computable or λ-definible)

Intensional Meaning versus Extensional Meaning

> *'Here we also use the phrase "Church-Turing thesis" to refer to the amalgamation of the two theses (these and others) where we identify all informal concepts of Definition 1.1*[†] *with one another we identify all the formal concepts of Definition 1.2*[‡]*, and their mathematical equivalents, with one another and suppress their intensional meanings.' [Soare 1996, p. 296]*

[†]Definition 1.1: A function is 'computable' (also called 'effectively calculable' or simply 'calculable') if it can be calculated by a finite mechanical procedure.

[‡]Definition 1.2: (i) A function is 'Turing computable' if it is definable by a Turing machine, as defined by Turing 1936.

Possible Refutations

Turing machine computability does not imply effective calculability

> *'A function is considered effectively computable if its value can be computed in an effective way in a finite number of steps, but there is no bound on the number of steps required for any given computation. Thus, the fact that there are effectively computable functions which may not be humanly computable has nothing to do with Church's thesis.' [Mendelson 1963, p. 202]*

Possible Refutations

Effective calculability does not imply Turing machine computability

From a Church's letter to Pepis (June 8, 1937):

> *'Therefore to discover a function which was effectively calculable but no general recursive would imply discovery of an utterly new principle of logic, not only never before formulated, but never before actually used in a mathematical proof...Moreover this new principle of logic must be of so strange, and presumably complicated,...I should be inclined to scrutinize the alleged effective applicability of the principle with considerable care.' [Sieg 1997, pp. 175–176]*

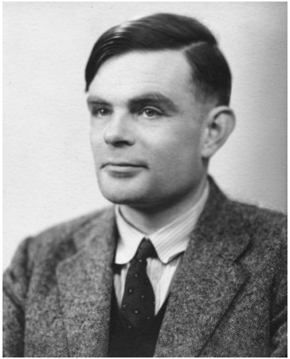
Alonzo Church: A Definition

'We now define the notion, already discussed, of an effectively calculable function of positive integers by identifying it with the notion of a recursive function of positive integers (or of a λ-definable function of positive integers).' [Church 1936, p. 356]

See also [Church 1935].

Alan Turing: A Definition

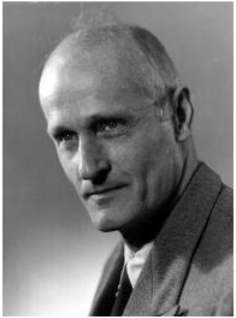
'The "computable" numbers[†] include all numbers which would naturally be regarded as computable.' [Turing 1936–1937, p. 249]



[†]The numbers whose decimal representation can be generating progressively by a Turing machine.

Stephen Kleene: Church's Thesis and Turing's Thesis

'The thesis of Church and Turing were not even called "thesis" at all until Kleene [1943, p. 60] referred to Church's "definition" as "Thesis I", and then in 1952 Kleene referred to "Church's Thesis" and "Turing's Thesis".' [Soare 1996, pp. 295–296]

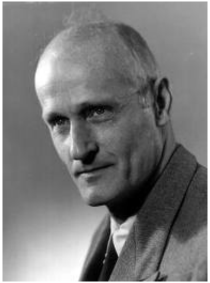


Stephen Kleene: The Church-Turing Thesis

Jay and Vergara [2004] point out the term 'Church-Turing thesis' was first named—but not defined—by Kleene [1974, p. 382].

Stephen Kleene: The Church-Turing Thesis

'The term "Church-Turing thesis" seems to have been first introduce by Kleene, with a small flourish of bias in favor of Church:' [Copeland 2002]



'So Turing's and Church's thesis are equivalent. We shall usually refer to them both as Church's thesis, or in connection with that one of its...version which deal with "Turing machines" as the Church-Turing thesis.' [Kleene 1967, p. 232]

Misunderstanding: Human Computers or Machines

Turing's analysis: Features of computations performed by human computers

- 'States of mind' $\Rightarrow$ Finite number of states

Misunderstanding: Human Computers or Machines

Turing's analysis: Features of computations performed by human computers

- 'States of mind' $\Rightarrow$ Finite number of states
- A human cannot reliably discriminate infinitely many symbols $\Rightarrow$ Finite alphabet

Misunderstanding: Human Computers or Machines

Turing's analysis: Features of computations performed by human computers

- 'States of mind' $\Rightarrow$ Finite number of states
- A human cannot reliably discriminate infinitely many symbols $\Rightarrow$ Finite alphabet
- Unlimited sheets of paper $\Rightarrow$ Unbounded tape

Misunderstanding: Human Computers or Machines

Turing's analysis: Features of computations performed by human computers

- 'States of mind' $\Rightarrow$ Finite number of states
- A human cannot reliably discriminate infinitely many symbols $\Rightarrow$ Finite alphabet
- Unlimited sheets of paper $\Rightarrow$ Unbounded tape
- The human read/write symbols on the paper $\Rightarrow$ Read/Write head

Misunderstanding: Human Computers or Machines

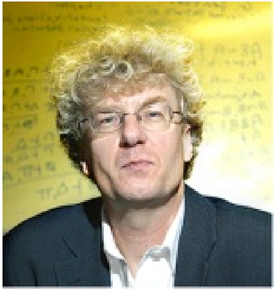
Turing's analysis: Features of computations performed by human computers

- 'States of mind' $\Rightarrow$ Finite number of states
- A human cannot reliably discriminate infinitely many symbols $\Rightarrow$ Finite alphabet
- Unlimited sheets of paper $\Rightarrow$ Unbounded tape
- The human read/write symbols on the paper $\Rightarrow$ Read/Write head
- Human's shift of attention form one part of the paper to another $\Rightarrow$ Displacement of the read/write head

Misunderstanding: Human Computers or Machines

A better version of the Church-Turing-Kleene thesis

'Any procedure than can be carried out by an idealised human clerk working mechanically with paper and pencil can also be carried out by a Turing machine.' [Copeland and Sylvan 1999].



Misunderstanding: Human Computers or Machines

Gandy's theses [Gandy 1980]

'Thesis P. A discrete deterministic mechanical device satisfies principles I-IV below.' [p. 126]

'Theorem. What can be calculated by a device satisfying principles I-IV is computable.' [p. 126]

'Thesis M. What can be calculated by a machine is Turing machine computable.' [p. 124]

Misunderstanding: Human Computers or Machines

Physical Church-Turing-Kleene thesis

> *'A function is computable by means of a physically possible computing device if and only if there is a Turing machine which computes it.' [Galton 2006, p. 95]*

Misunderstanding: Human Computers or Machines

Current situation

- At the moment, it does not exists a refutation to the Church-Turing-Kleene thesis.

Misunderstanding: Human Computers or Machines

Current situation

- At the moment, it does not exists a refutation to the Church-Turing-Kleene thesis.
- The hypercomputation models refute the theoretical version of the thesis M.

Misunderstanding: Human Computers or Machines

Current situation

- At the moment, it does not exists a refutation to the Church-Turing-Kleene thesis.
- The hypercomputation models refute the theoretical version of the thesis M.
- Open problem: the refutation of the realizable version of the thesis M (i.e. the physical Church-Turing thesis).

Misunderstanding: Functions or Number-Theoretic Functions

Definition

Let A be a type and let f and $\perp$ be a terminating and a non-terminating function from a to a, respectively. Plotkin [1977] **`parallelOr`** function has the following behaviour:

$$\begin{aligned}
&\text{parallelOr} : (a \to a) \to (a \to a) \to a \to a \\
&\text{parallelOr}\, f \perp = f \\
&\text{parallelOr} \perp f = f \\
&\text{parallelOr} \perp \perp = \perp
\end{aligned}$$

Misunderstanding: Functions or Number-Theoretic Functions

Definition

Let A be a type and let f and $\perp$ be a terminating and a non-terminating function from a to a, respectively. Plotkin [1977] **parallelOr** function has the following behaviour:

$$\begin{aligned}
&\text{parallelOr} : (a \to a) \to (a \to a) \to a \to a \\
&\text{parallelOr}\, f\, \perp = f \\
&\text{parallelOr}\, \perp\, f = f \\
&\text{parallelOr}\, \perp\, \perp = \perp
\end{aligned}$$

Theorem

The `parallelOr` function is an effectively calculable function which is not λ-definable [Plotkin 1977]. See, also, [Turner 2006].

Misunderstanding: Functions or Number-Theoretic Functions

Definition

Let Δ be the set of λ-terms, let $\equiv$ be the syntactic identity on λ-terms and let M and N be two combinators in β-normal form. **Church's $\boldsymbol{\delta}$** function is defined by

$$\delta : \Delta \to \Delta \to \Delta$$

$$\delta M N := \begin{cases} \text{true}, & \text{if } M \equiv N; \\ \text{false}, & \text{if } M \not\equiv N. \end{cases}$$

Theorem

Church's δ function is not λ-definable [Barendregt 2004, Corollary 20.3.3, p. 520].

Misunderstanding: Functions or Number-Theoretic Functions

Extensions of λ-calculus

Jay and Vergara [2017] wrote (emphasis is ours):

> *'For over fifteen years, the lead author has been developing calculi that are more expressive than λ-calculus, beginning with the constructor calculus [8], then pattern calculus [2,7,3], SF-calculus [6] and now λSF-calculus [5]...*
>
> *[The] λSF-calculus is able to query programs expressed as λ-abstractions, as well as combinators, something that is beyond pure λ-calculus.*
>
> *In particular, we have proved (and verified in Coq [4]) that equality of closed normal forms is definable within λSF-calculus.'*

Misunderstanding: Functions or Number-Theoretic Functions

Extensions of λ-calculus

Jay and Vergara [2017] also stated the following corollaries:

(i) Church's δ is λSF-definable.

(ii) Church's δ is λ-definable.

(iii) Church's δ is not λ-definable.

Misunderstanding: Functions or Number-Theoretic Functions

Question

Do Plotkin's `parallelOr` function or Church's δ function—which are effectively calculable functions but they are not λ-definable functions—contradict the Church-Turing-Kleene thesis?

Misunderstanding: Functions or Number-Theoretic Functions

Question

Do Plotkin's `parallelOr` function or Church's δ function—which are effectively calculable functions but they are not λ-definable functions—contradict the Church-Turing-Kleene thesis?

Answer. No! But we need a better version of the Church-Turing-Kleene thesis.

Discussion

Definition

A **number-theoretic function** is a function whose signature is

$$\mathbb{N}^k \to \mathbb{N}, \text{ with } k \in \mathbb{N}.$$

Discussion

Definition

A **number-theoretic function** is a function whose signature is

$$\mathbb{N}^k \to \mathbb{N}, \text{ with } k \in \mathbb{N}.$$

Theorem (corrected version)

The following sets are coextensive:

(i) the λ-definable number-theoretic functions and

(ii) the number-theoretic functions computable by a Turing machine

Discussion

A better version of the Church-Turing-Kleene thesis

We should write the Church-Turing-Kleene thesis as:

Any number-theoretic function than can be computed by an idealised human clerk working mechanically with paper and pencil can also be computed by a Turing machine.

Discussion

A better version of the Church-Turing-Kleene thesis

We should write the Church-Turing-Kleene thesis as:

Any number-theoretic function than can be computed by an idealised human clerk working mechanically with paper and pencil can also be computed by a Turing machine.

Observation

Jay and Vergara [2004, 2017] also negatively answer the question under discussion stating other versions of the Church-Turing-Kleene thesis.

Bonus Slide

Higher-order computability

There are various notions of computability in higher-order settings (see, e.g. [Longley and Normann 2015]).

References

Barendregt, H. P. [1981] (2004). The Lambda Calculus. Its Syntax and Semantics. Revised edition, 6th impression. Vol. 103. Studies in Logic and the Foundations of Mathematics. Elsevier (cit. on p. 27).

Church, A. (1935). An Unsolvable Problem of Elementary Number Theory. Preliminar Report (Abstract). Bulletin of the American Mathematical Society 41.5, pp. 332–333. DOI: 10.1090/S0002-9904-1935-06102-6 (cit. on p. 9).

— (1936). An Unsolvable Problem of Elementary Number Theory. American Journal of Mathematics 58.2, pp. 345–363. DOI: 10.2307/2371045 (cit. on p. 9).

Copeland, B. J. (19th Aug. 2002). The Church-Turing Thesis. Stanford Encyclopedia of Philosophy (cit. on p. 13).

Copeland, B. J. and Sylvan, R. (1999). Beyond the Universal Turing Machine. Australasian Journal of Philosophy 77.1, pp. 44–66. DOI: 10.1080/00048409912348801 (cit. on p. 19).

Galton, A. (2006). The Church-Turing Thesis: Still Valid after All These Years? Applied Mathematics and Computation 178.1, pp. 93–102. DOI: 10.1016/j.amc.2005.09.086 (cit. on p. 21).

References

Gandy, R. (1980). Church's Thesis and Principles for Mechanisms. In: The Kleene Symposium. Ed. by Barwise, J., Keisler, H. J. and Kunen, K. Vol. 101. Studies in Logic and the Foundations of Mathematics. North-Holland Publishing Company, pp. 123–148. DOI: 10.1016/S0049-237X(08)71257-6 (cit. on p. 20).

Jay, B. and Vergara, J. (2004). Confusion in the Church-Turing Thesis. Draft version. URL: https://arxiv.org/abs/1410.7103 (cit. on pp. 12, 34, 35).

— (2017). Conflicting Accounts of -Definability. Journal of Logical and Algebraic Methods in Programming 87, pp. 1–3. DOI: 10.1016/j.jlamp.2016.11.001 (cit. on pp. 28, 29, 34, 35).

Kleene, S. C. (1943). Recursive Predicates and Quantifiers. Transactions of the American Mathematical Society 53.1, pp. 41–73. DOI: 10.1090/S0002-9947-1943-0007371-8 (cit. on p. 11).

Kleene, S. C. (1967). Mathematical Logic. John Wiley & Sons (cit. on p. 13).

Kleene, S. C. [1952] (1974). Introduction to Metamathematics. Seventh reprint. North-Holland (cit. on p. 12).

References

Longley, J. R. and Normann, D. (2015). Higher-Order Computability. Theory and Applications of Computability. Springer. DOI: 10.1007/978-3-662-47992-6 (cit. on p. 36).

Mendelson, E. (1963). On Some Recent Criticism of Church's Thesis. Notre Dame Journal of Formal Logic 4.3, pp. 201–205. DOI: 10.1305/ndjfl/1093957577 (cit. on p. 7).

Plotkin, G. D. (1977). LCF Considered as a Programming Language. Theoretical Computer Science 5.3, pp. 223–255. DOI: 10.1016/0304-3975(77)90044-5 (cit. on pp. 25, 26).

Sieg, W. (1997). Step by Recursive Step: Church's Analysis of Effective Calculability. The Bulletin of Symbolic Logic 3.2, pp. 154–180. DOI: 10.2307/421012 (cit. on p. 8).

Soare, R. I. (1996). Computability and Recursion. The Bulletin of Symbolic Logic 2.3, pp. 284–321. DOI: 10.2307/420992 (cit. on pp. 6, 11).

Turing, A. M. (1936–1937). On Computable Numbers, with an Application to the Entscheidungsproblem. Proceeding of the London Mathematical Society s2-42, pp. 230–265. DOI: 10.1112/plms/s2-42.1.230 (cit. on p. 10).

References

Turner, D. (2006). Church's Thesis and Functional Programming. In: Church's Thesis After 70 Years. Ed. by Olszewski, A., Woleński, J. and Janusz, R. Ontos Verlag, pp. 518–544. DOI: 10.1515/9783110325461.518 (cit. on pp. 25, 26).