Automata and Formal Languages - CM0081
The Church-Turing Thesis

Andrés Sicard-Ramírez

Universidad EAFIT

Semester 2019-1
Common Versions of the Church-Turing Thesis

Theorem (first version)

The following sets are coextensive:

i) the \( \lambda \)-definable functions,

ii) the functions computable by a Turing machine and

iii) the general recursive functions.
Common Versions of the Church-Turing Thesis

“A function is computable (effectively calculable) if and only if there is a Turing machine which computes it.” [Galton 2006, p. 94]
Common Versions of the Church-Turing Thesis

“A function is *computable (effectively calculable)* if and only if there is a *Turing machine* which computes it.” [Galton 2006, p. 94]

“The unprovable assumption that any general way to compute will allow us compute only the partial-recursive functions (or equivalently, what Turing machines or modern-day computers can compute) is know as *Church’s hypothesis* or the *Church-Turing thesis.*” [Hopcroft, Motwani and Ullman 2007, p. 236]
“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 $\lambda$-definable function of positive integers). This definition is thought to be justified by considerations which follow, so far as positive justification can ever be obtained for the selection of a formal definition to correspond to intuitive notion.” [Church 1936, p. 356]

See also [Church 1935].
“The ‘computable’ numbers† include all numbers which would naturally be regarded as computable.” [Turing 1936, p. 249]

†The number whose decimal representation can be generating progressively by a Turing machine.
“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]
Jay and Vergara [2004] point out the term 'Church-Turing thesis' was first named—but not defined—in Kleene [1952, p. 382].
“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]
“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

Idea: Turing machine computability $\not\Rightarrow$ 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

Idea: Effective calculability $\Rightarrow$ Turing machine computability
(the interesting one!)

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]
A Misunderstanding: Human Computers or Machines

Turing’ analysis: Features of computations performed by human computers

- “States of mind” $\Rightarrow$ Finite number of states
A Misunderstanding: Human Computers or Machines

Turing’ 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
A 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
A Misunderstanding: Human Computers or Machines

Turing’ analysis: Features of computations performed by human computers

- “States of mind” ⇒ Finite number of states
- A human cannot reliably discriminate infinitely many symbols ⇒ Finite alphabet
- Unlimited sheets of paper ⇒ Unbounded tape
- The human read/write symbols on the paper ⇒ Read/Write head
A Misunderstanding: Human Computers or Machines

Turing’ 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
A Misunderstanding: Human Computers or Machines

A better version of the Church-Turing 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].
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)
A 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)
A Misunderstanding: Human Computers or Machines

Physical Church-Turing 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].
A Misunderstanding: Human Computers or Machines

Current Situation

- At the moment, it does not exist a refutation to the Church-Turing thesis.
Current Situation

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

- The hypercomputation models refute the theoretical version of the thesis M.
Current Situation

- At the moment, it does not exist a refutation to the Church-Turing 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).
Other Misunderstanding: Functions or Number-Theoretic Functions

Definition

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

\[
\begin{align*}
p\text{Or} & : (a \to a) \to (a \to a) \to a \to a \\
p\text{Or} f \bot & = f \\
p\text{Or} \bot f & = f \\
p\text{Or} \bot \bot & = \bot 
\end{align*}
\]
Other Misunderstanding: Functions or Number-Theoretic Functions

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

$$pOr :: (a \to a) \to (a \to a) \to a \to a$$
$$pOr \ f \ \bot = f$$
$$pOr \ \bot \ f = f$$
$$pOr \ \bot \ \bot = \bot$$

Theorem
The parallel-or function is an effectively calculable function which is not $\lambda$-definable [Plotkin 1977]. See, also, Turner [2006].
Other Misunderstanding: Functions or Number-Theoretic Functions

Definition
Let \( M \) and \( N \) be combinators in \( \beta \)-normal form. Church’s \( \delta \) function is defined by

\[
\delta M N = \begin{cases} 
  \text{true,} & \text{if } M \equiv N; \\
  \text{true,} & \text{if } M \not\equiv N.
\end{cases}
\]

Theorem
Church’s \( \delta \) function is not \( \lambda \)-definable [Barendregt 2004, Corollary 20.3.3, p. 520].
Other 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.”
Other Misunderstanding: Functions or Number-Theoretic Functions

Extensions of $\lambda$-calculus

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

1. Church’s $\delta$ is $\lambda SF$-definable.
2. Church’s $\delta$ is $\lambda$-definable.
3. Church’s $\delta$ is not $\lambda$-definable.
Other Misunderstanding: Functions or Number-Theoretic Functions

Question
Do Plotkin’s parallel-or function or Church’s $\delta$ function—which are effectively calculable functions but they are not $\lambda$-definable functions—contradict the Church-Turing thesis?
Other Misunderstanding: Functions or Number-Theoretic Functions

Question
Do Plotkin’s parallel-or function or Church’s $\delta$ function—which are effectively calculable functions but they are not $\lambda$-definable functions—contradict the Church-Turing thesis?

A/ No! But we need a better version of the Church-Turing thesis.
Discussion

Definition

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

\[ \mathbb{N}^k \rightarrow \mathbb{N}, \text{ with } k \in \mathbb{N}. \]
Discussion

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

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

Theorem

corrected version The following sets are coextensive:

i) the $\lambda$-definable number-theoretic functions,

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

iii) the general recursive functions.

Remark
The above theorem is historically precise as pointed out in [Jay and
Vergara 2004].
Discussion

A better version of the Church-Turing thesis

We should write the Church-Turing 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.
A better version of the Church-Turing thesis

We should write the Church-Turing 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.

Remark

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

A (accepted) definition of higher-type computability is an open problem (see, for example, [Longley 2003]).
References


References


