Church-Turing Completeness: Syntax and Semantics

Robert Irving Soare
Paul Snowden Russell
Distinguished Service Professor

The University of Chicago

The Incomputable
Chicheley Hall
Tuesday, June 12, 2012

“... it appears that there is no way of finding the general criterion for deciding whether or not a well-formed formula is provable. (We cannot at the moment establish this. Indeed, we have no clue as to how such a proof of undecidability would go.) ... the undecidability is even the *conditio sine qua non* for the contemporary practice of mathematics, using as it does heuristic methods to make any sense.
"... it appears that there is no way of finding the general criterion for deciding whether or not a well-formed formula is provable. (We cannot at the moment establish this. Indeed, we have no clue as to how such a proof of undecidability would go.) ... the undecidability is even the conditio sine qua non for the contemporary practice of mathematics, using as it does heuristic methods to make any sense.

The very day on which the undecidability does not obtain anymore, mathematics as we now understand it would cease to exist; it would be replaced by an absolutely mechanical prescription (eine absolut mechanische Vorschrift), by means of which anyone could decide the provability or unprovability of any sentence.

Thus, we have to take the position: it is generally undecidable, whether a given well-formed formula is provable or not"
Gödel’s Opinion [1934]

Gödel had considered the question of characterizing the calculable functions in [1934] when he wrote:

“[Primitive] recursive functions have the important property that, for each given set of values for the arguments, the value of the function can be computed by a finite procedure.”

Footnote 3.

“The converse seems to be true, if, besides recursion according to scheme (V) [primitive recursion], recursions of other forms (e.g., with respect to two variables simultaneously) are admitted. This cannot be proved, since the notion of finite computation is not defined, but it serves as a heuristic principle.”
Gödel’s Opinion [1934]

Gödel had considered the question of characterizing the calculable functions in [1934] when he wrote:

“[Primitive] recursive functions have the important property that, for each given set of values for the arguments, the value of the function can be computed by a finite procedure\(^3\).”
Gödel’s Opinion [1934]

Gödel had considered the question of characterizing the calculable functions in [1934] when he wrote:

“[Primitive] recursive functions have the important property that, for each given set of values for the arguments, the value of the function can be computed by a finite procedure.”

Footnote 3.
“The converse seems to be true, if, besides recursion according to scheme (V) [primitive recursion], recursions of other forms (e.g., with respect to two variables simultaneously) are admitted. This cannot be proved, since the notion of finite computation is not defined, but it serves as a heuristic principle.”
Three Fundamental Topics on Turing and Computability

1. Church-Turing completeness.
Three Fundamental Topics on Turing and Computability

1. Church-Turing completeness.
2. Turing oracle machine [1939, §4].
Three Fundamental Topics on Turing and Computability

1. Church-Turing completeness.

2. Turing oracle machine [1939, §4].

3. Formalism and informalism (intuition) in computability theory.
SYNTAX and Extensional Properties of Computability Theory

Extensional (Syntactical) Characterization
SYNTAX and Extensional Properties of Computability Theory

Extensional (Syntactical) Characterization

**Confluence Theorem.** Kleene-Church [1936] and Turing [1937].

\[ f \text{ recursive} \iff f \lambda\text{-definable} \iff f \text{ Turing computable}. \]
SYNTAX and Extensional Properties of Computability Theory

Extensional (Syntactical) Characterization

**Confluence Theorem.** Kleene-Church [1936] and Turing [1937].

\[
f \text{ recursive } \iff f \text{ } \lambda\text{-definable } \iff f \text{ Turing computable.}
\]

SEMANTICS of Computability Theory

**Defn (1930’s).** A function \( f \) is (effectively) calculable, i.e., intuitively computable, if there is a finite procedure (algorithm) to calculate it.
For \( \sigma \) a sentence of first order logic,

\[ \sigma \text{ valid} \implies \sigma \text{ is provable.} \]
For $\sigma$ a sentence of first order logic,

$$\sigma \text{ valid } \implies \sigma \text{ is provable.}$$

Want Completeness for Computability i.e.

$$f \text{ effectively calculable } \implies f \text{ Turing computable (recursive).}$$
For $\sigma$ a sentence of first order logic,

$$\sigma \text{ valid } \implies \sigma \text{ is provable.}$$

Want Completeness for Computability i.e.

$$f \text{ effectively calculable } \implies f \text{ Turing computable (recursive).}$$

= Church-Turing Thesis? Not exactly.
Proof [1936] of Church’s Thesis (Completeness):
Every effectively calculable function is recursive.

\[ f \text{ eff. calculable} \implies \vdash_L f \implies f \text{ recursive.} \]

\[ \vdash_L f \] means \textbf{reckonable}. Hilbert Bernays 1934.
Proof [1936] of Church’s Thesis (Completeness):
Every effectively calculable function is recursive.

\[ f \text{ eff. calculable} \implies \vdash_L f \implies f \text{ recursive.} \]

\[ \vdash_L f \] means \textit{reckonable}. Hilbert Bernays 1934.
Frege 1879.
Zermelo, 1904, 1908
Russell-Whitehead, 1910, 1912, 1913.
Bernays 1922–1934
Herbrand 1928-1931
Hilbert 1900–1935
Hilbert and Ackermann, 1928
Gödel 1931, 1934 (Recursive Functions)
Kleene 1936 (Normal Form, $T$-predicate)
Gandy And Sieg Say:

The fatal weakness in Church’s argument was the core assumption that the atomic steps were stepwise recursive. Gandy [1988] and Sieg [1994] brought out this weakness. Sieg wrote, “. . . this core does not provide a convincing analysis: steps taken in a calculus must be of a restricted character and they are assumed, for example by Church, without argument to be recursive.”

“It is precisely here that we encounter the major stumbling block for Church’s analysis, and that stumbling block was quite clearly seen by Church,” who wrote that without this assumption it is difficult to see how the notion of a system of logic can be given any exact meaning at all. It is exactly this stumbling block which Turing overcame by a totally new approach.
Turing Completeness

Turing’s Thesis

Turing’s Soundness Theorem (Trivial)
If \( f \) is Turing computable it is effectively calculable.

Turing [1936] Completeness
If \( f \) is effectively calculable, then \( f \) is Turing computable.

Proof. Turing [1936: §9]

Calculable \( \implies \) computorable \( \implies \) Turing computable

Computorable: Precursor to Turing machine
Gandy [1988], Sieg [1994].
Turing proposed a number of simple operations “so elementary that it is not easy to imagine them further subdivided.”

- Divide the work space into squares. May assume 1 dimensional.
- Finitely many symbols. Each square contains one symbol.
- Finitely many states (of mind).
- Action of the machine determined by the present state and the squares observed.

Squares observed are bounded by $B$, say $B = 1$.

- Reading head examines one symbol in one square,
- May assume the machine moves to only squares within a radius of $L$ (of current square). May assume $L = 1$.
- Machine may print a symbol in the current square, change state, and move to adjacent square.
Robin Gandy  Professor at Manchester:

“Turing’s analysis does much more than provide an argument for” [for the thesis] “it proves a theorem.”

“Turing’s analysis makes no reference whatsoever to calculating machines. Turing machines appear as a result, a codification, of his analysis of calculations by humans.”
Kurt Gödel:

“That this really is the correct definition of mechanical computability was established beyond any doubt by Turing.”

“But I was completely convinced only by Turing’s paper.”
Gödel In 1964 postscript to [1934]:

In consequence of later advances, in particular of the fact that, due to A.M. Turing’s work, a precise and unquestionably adequate definition of the general concept of formal system can now be given, the existence of undecidable arithmetical propositions and the non-demonstrability of the consistency of a system in the same system can now be proved rigorously for every consistent formal system containing a certain amount of finitary number theory.

Turing’s work gives an analysis of the concept of mechanical procedure, alias algorithm or computation procedure or finite combinatorial procedure. This concept is shown to be equivalent with that of a Turing machine.
—Gödel: 1946 Princeton Bicentennial, Godel [1946: 64]

... one [Turing] has for the first time succeeded in giving an absolute definition of an interesting epistemological notion, i.e., one not depending on the formalism chosen.

... For the concept of computability, however, although it is merely a special kind of demonstrability or decidability, the situation is different. By a kind of miracle it is not necessary to distinguish orders, and the diagonal procedure does not lead outside the defined notion.
“The greatest improvement was made possible through the precise definition of the concept of finite procedure, . . . This concept, . . . is equivalent to the concept of a “computable function of integers” . . .

The most satisfactory way, in my opinion, is that of reducing the concept of finite procedure to that of a machine with a finite number of parts, as has been done by the British mathematician Turing.”
Church: 

“Of the three different notions: computability by a Turing machine, general recursiveness of Herbrand-Gödel-Kleene, and λ-definability.”

“The first has the advantage of making the identification with effectiveness in the ordinary (not explicitly defined) sense evident immediately—i.e., without the necessity of proving preliminary theorems.”
Church [1937a] on Turing [1936].

...in particular, a human calculator, provided with pencil and paper and explicit instructions, can be regarded as a kind of Turing machine. It is thus immediately clear that computability, so defined, can be identified with ... the notion of effectiveness as it appears in certain mathematical problems (various forms of the Entscheidungsproblem, various problems to find complete sets of invariants in topology, group theory, etc., and in general any problem which concerns the discovery of an algorithm).
Church-Turing Completeness

Church [1937b] on Post [1936].

The author . . . [Post] proposes a definition of “finite 1-process” which is equivalent, to computation by a Turing machine. He does not, however, regard his formulation as certainly to be identified with effectiveness in the ordinary sense, but takes this identification as a “working hypothesis” in need of continual verification. To define effectiveness as computability by an arbitrary machine, subject to restrictions of finiteness, would seem to be an adequate representation of the ordinary notion, and if this is done the need for a working hypothesis disappears.
Undecidable and decidable problems in mathematics

Professor Angus MacIntyre

Gresham College Lecture, May 17, 2011
Queen Mary College, University of London
President of the London Mathematical Society

Turing [1936] accomplished:

- exact definition of undecidability of math problems
- astoundingly simple
- one of best papers in Proc. London Math. Soc. in the century.
Defn

**Thesis.** A proposition stated or put forward for consideration, especially one to be discussed and proved or to be maintained against objections. not a topic, **not a fact**, not an opinion.

*e.g.* He vigorously defended his thesis on the causes of war.

synonyms: theory, contention, proposal.
Option 1: Turing completeness is a fact.

Gödel, “That this really is the correct definition of mechanical computability was established beyond any doubt by Turing.”

Gandy, “Turing’s analysis does much more than provide an argument for” [for the thesis] “it proves a theorem.”

Proponents:

Church
Turing
Gödel
Post
Wilfried Sieg
Robin Gandy
Angus Macintyre
Option 2: Church-Turing completeness is a \textit{thesis}.

Kleene [1952]:

1. Make Church’s Thesis an \textit{unproved hypothesis (conjecture)}.
2. We cannot prove Church’s Thesis (so will not try).
3. Confluence. The syntactic characterizations are equal extensionally.
4. A wide variety of effectively calculable functions has been studied. All are recursive.
5. We have failed to find a counterexample.
Gödel on Non-Mechanical Procedures

Regarding the possibility of other non-mechanical procedures, Gödel [1964] wrote in Davis [1965: 72],

*Note that the question of whether there exist non-mechanical procedures not equivalent with any algorithm, has nothing whatsoever to do with the adequacy of the definition of “formal system” and of “mechanical procedure.”*
[1936] Neither Church nor Turing called it a thesis.

Kleene [1943] Kleene called it Thesis I.


Informal Arguments in the main books 1950–1970: Kleene [1952], Rogers [1967],

1. Make Church’s Thesis an unproved hypothesis (conjecture).
2. We cannot prove Church’s Thesis (so will not try).
3. Confluence. The characterizations are equal extensionally.
4. A wide variety of effectively calculable functions has been studied. All are recursive.
5. We have failed to find a counterexample.
Is There an Undecidable Computably Enumerable (C.E.) Set?

Define $K = \{ e : e \in W_e \}$.

- $K$ is computably enumerable.
- $K$ is not recursive.
- $K$ is not Turing computable.

Question. Is $K$ Undecidable?

Needed for undecidability of:

1. Entscheidungsproblem
2. Hilbert’s 10th Problem
3. Word problem for finitely presented groups
4. Differential geometry
Turing’s Oracle Machine


“Let us suppose we are supplied with some unspecified means of solving number-theoretic problems; a kind of oracle as it were. . . . this oracle . . . cannot be a machine.

With the help of the oracle we could form a new kind of machine (call them o-machines), having as one of its fundamental processes that of solving a given number-theoretic problem.”
Let $\Phi^A_e(x)$ denote $o$-machine with G"odel number $e$ and with $A$ on the oracle tape. Define the halting set

$$K^A = \{ e : \Phi^A_e(e) \text{ halts} \}. \quad (1)$$

**Theorem**


Turing [1939]:

“Given any one of these machines we may ask the question whether it prints an infinity of figures 0 or 1; I assert that this class of problem is not number-theoretic (namely not $\Pi_2$).”

Let $A = \{ e : (\forall x)(\exists y > x) \left[ \varphi_e(y) = 1 \right] \}$.

Then $A$ is $\Pi_2$-complete. Hence, $K^A$ is not $\Pi_2$. 
Applications of o-Machines

- **Defn. Turing Reducibility (Turing 1936, Post, 1944).** \( B = \Phi^A_e \)

  Turing reducibility, degrees of unsolvability, information content, etc.

- **Defn.** An **interactive** computing process is one which interacts with its environment,

The First Wave of Formalism

First Wave of Formalism

Normal Form Theorem, Kleene, 1936

\[ \psi_e(x) = U(\mu y \ T(e, x, y)) \]

First Wave of Informalism and Intuition

Write \( \varphi_{e,s}(x) = z \) if \( P_e \) with input \( x \) halts in \( \leq s \) steps with output \( z \).

Post-Turing Normal Form.

\[ \text{PT}(e, x, s) \text{ iff } (\exists z \leq s)[ \varphi_{e,s}(x) = z ] \.]
Informal Approach of Emil Post 1944

“That mathematicians generally are oblivious to the importance of this work of Gödel, Church, Turing, Kleene, Rosser and others is in part due to the forbidding, diverse and alien formalisms.”

We have obtained formal proofs of all the consequently mathematical theorems here [1944] developed informally. Yet the real mathematics involved must lie in the informal development. For in every instance transforming it into the formal proof turned out to be a routine chore.”
Second Period of Formalism 1954–1965

1954: Turing and Post both died; Intuition receded

Thm. Kleene Post [1954].

\[(\exists A \leq_T \emptyset')(\exists B \leq_T \emptyset') \left[ A \not\leq_T B \land B \not\leq_T A \right].\]

Proof. Finite extension of strings.

Given \(\sigma_s \prec A\) and \(\tau_s \prec B\). Let \(n = |\sigma_s|\). Ask the \(\emptyset'\)-oracle,

\[(\exists t)(\exists \rho) \left[ \rho \succ \tau_s \land \Phi_{e,t}^\rho (n) \downarrow \right].\]
Kleene-Post Proof of Same Theorem

Friedberg [1957] Finite Injury

Sacks book on degrees [1963]
Second Period of Informalism and Intuition 1967, 1970

Hartley Rogers, book 1967

Papers by Lachlan:
1970 Games
1973 Topology of Priority Arguments
1975 Trees
[Soare, 2012a] = [Soare, CTA]

[Soare, 2012b]

[Soare, 2012c]
[Soare, 2012d]

[Soare, 2012e]

[Soare, 2012f]