Proving the Church-Turing Thesis?

Kerry Ojakian[1]

[1]SQIG/IT Lisbon and IST, Portugal

Logic Seminar 2008

Outline

Outline

The Church-Turing Thesis.

The set of calculable functions =
The set of (Turing Machine) computable functions

Impossible to prove?

LHS: An informal notion
RHS: A formal notion
Thus mathematical proof impossible (Standard view, see Folina)

Against standard view (see Mendelson, Black):

- We can already prove something (the easy direction)
- Maybe full proof possible!

Impossible to prove?

LHS: An informal notion
RHS: A formal notion
Thus mathematical proof impossible (Standard view, see Folina)

Against standard view (see Mendelson, Black):

- We can already prove something (the easy direction)
- Maybe full proof possible!

Axiomatize CT in "some way"

Alternative way to "prove" the CT thesis:

1. Find mathematically precise axioms for calculable
2. Prove that functions satisfying these axioms = TM-computable.

Informal Claim: The interest of the above approach = The interest of the first step

Axiomatize CT in "some way"

Alternative way to "prove" the CT thesis:

1. Find mathematically precise axioms for calculable
2. Prove that functions satisfying these axioms = TM-computable.

Informal Claim: The interest of the above approach = The interest of the first step

Outline

Definition of ASM

Definition

An Arithmetic ASM is specified by a finite set of dynamic function symbols, along with a finite program of updates.

Example (Addition):

- Static Function symbols: 0, S, $\bot$
- Dynamic Function symbols: C, in_1, in_2, out

$$\begin{array}{l} \textit{if } out = \bot \textit{ then } out := in_1 \\ \textit{if } C = \bot \textit{ then } C := 0 \\ \textit{if } C \neq in_2 \wedge C \neq \bot \textit{ then } out := out + 1 \\ \textit{if } C \neq in_2 \wedge C \neq \bot \textit{ then } C := C + 1 \end{array}$$

Proving Church-Turing via ASM?

"Proof" of CT in two steps (Boker, Dershowitz, Gurevich):

1. Axiomatize calculable by ASM-computability.
2. Prove that ASM-computability = TM-computable.

Step 1: Argument similar to Turing.
Step 2: Straightforward.

Recall Informal Claim:

> *An axiomatization of CT is only as interesting as the first step.*

Question:

> *Does step 1 provide an axiomatization of calculable that is more interesting than Turing machines or other models of computation?*

Proving Church-Turing via ASM?

"Proof" of CT in two steps (Boker, Dershowitz, Gurevich):

1. Axiomatize calculable by ASM-computability.
2. Prove that ASM-computability = TM-computable.

Step 1: Argument similar to Turing.
Step 2: Straightforward.

Recall Informal Claim:

> *An axiomatization of CT is only as interesting as the first step.*

Question:

> *Does step 1 provide an axiomatization of calculable that is more interesting than Turing machines or other models of computation?*

The good and the bad of the ASM approach ...

The good ...

- The ASM model allows a more flexible/general definition of states and the domain.
- The ASM model allows us to more easily add or subtract "axioms".

The bad ...

It is fundamentally "device-dependent"!

The good and the bad of the ASM approach ...

The good ...

- The ASM model allows a more flexible/general definition of states and the domain.
- The ASM model allows us to more easily add or subtract "axioms".

The bad ...

It is fundamentally "device-dependent"!

Criticism of ASM and related approaches

Shore says:

> *"Prove" the Church-Turing thesis by finding intuitively obvious or at least clearly acceptable properties of computation*

However he goes on to say:

> *Perhaps the question is whether we can be sufficiently precise about what we mean by computation without reference to the method of carrying out the computation so as to give a more general or more convincing argument independent of the physical or logical implementation.*

Criticism of ASM and related approaches

Shore says:

> *"Prove" the Church-Turing thesis by finding intuitively obvious or at least clearly acceptable properties of computation*

However he goes on to say:

> *Perhaps the question is whether we can be sufficiently precise about what we mean by computation without reference to the method of carrying out the computation so as to give a more general or more convincing argument independent of the physical or logical implementation.*

Other device-dependent approaches

Criticism extends to other device-dependent axiomatizations:

- Turing-Machines
- Recursive Functions
- Representable in Arithmetic
- Gandy's Machines (1980)

Outline

The device-independent approach

Definition

(First Try) A definition of calculable is device-independent if it is not of the form: f is calculable iff there is a finite device M such that M calculates f.

Example: f is TM-computable iff $\exists M \; \forall x \; M(x) = f(x)$

Loose Claim: All existing formal definitions of computable are naturally written as predicates of complexity Σ^0_3.

Definition

(Second Try) A definition of calculable is device-independent iff its complexity is below Σ^0_3.

The device-independent approach

Definition

(First Try) A definition of calculable is device-independent if it is not of the form: f is calculable iff there is a finite device M such that M calculates f.

Example: f is TM-computable iff $\exists M\ \forall x\ M(x) = f(x)$

Loose Claim: All existing formal definitions of computable are naturally written as predicates of complexity Σ^0_3.

Definition

(Second Try) A definition of calculable is device-independent iff its complexity is below Σ^0_3.

Examples of device-independent axioms

- The set of calculable functions is countable.
- The set of calculable functions contains a universal function for itself.
- The set of calculable functions satisfies T (where T is some typical theorem of computability theory).

Many reasonable axioms not even arithmetic!

Definition

(Third Try) If a definition is below Σ^0_3 then it is device-independent.

First Question: Is there a definition below Σ^0_3?
NO!

Examples of device-independent axioms

- The set of calculable functions is countable.
- The set of calculable functions contains a universal function for itself.
- The set of calculable functions satisfies T (where T is some typical theorem of computability theory).

Many reasonable axioms not even arithmetic!

Definition

(Third Try) If a definition is below Σ_3^0 then it is device-independent.

First Question: Is there a definition below Σ_3^0?

NO!

Examples of device-independent axioms

- The set of calculable functions is countable.
- The set of calculable functions contains a universal function for itself.
- The set of calculable functions satisfies T (where T is some typical theorem of computability theory).

Many reasonable axioms not even arithmetic!

Definition

(Third Try) If a definition is below Σ^0_3 then it is device-independent.

First Question: Is there a definition below Σ^0_3?
NO!

Computable functions in the arithmetic hierarchy

Let $\mathcal{C} = \{f \in \omega^\omega \mid f \text{ is computable } \}$

Theorem

(Shoenfield 1958) $\mathcal{C}$ is in $\Sigma^0_3 - \Pi^0_3$.

Thus, no device-independent definition in the Arithmetic Hierarchy.

What about outside the Arithmetic Hierarchy?

Computable functions in the arithmetic hierarchy

Let $\mathcal{C} = \{f \in \omega^\omega \mid f \text{ is computable } \}$

Theorem

(Shoenfield 1958) $\mathcal{C}$ is in $\Sigma^0_3 - \Pi^0_3$.

Thus, no device-independent definition in the Arithmetic Hierarchy.

What about outside the Arithmetic Hierarchy?

Is Axiomatizing the CT "impossible"?!

- Huge difficulty: Most Properties Relativize.
- Extend definition of device-independent to higher order quantifiers?
  Problem: We have the entire Arithmetic Hierarchy as soon as we have higher order quantifiers (using the standard approach).

Programme:

1. Develop a more finely stratified hierarchy.
2. Define device-independent as "below Σ^0_3"
3. Search for upper and lower bounds on $\mathcal{C}$ in this hierarchy.

Is Axiomatizing the CT "impossible"?!

- Huge difficulty: Most Properties Relativize.
- Extend definition of device-independent to higher order quantifiers?
  Problem: We have the entire Arithmetic Hierarchy as soon as we have higher order quantifiers (using the standard approach).

Programme:

1. Develop a more finely stratified hierarchy.
2. Define device-independent as "below Σ_3^0"
3. Search for upper and lower bounds on $\mathcal{C}$ in this hierarchy.

Is Axiomatizing the CT "impossible"?!

- Huge difficulty: Most Properties Relativize.
- Extend definition of device-independent to higher order quantifiers?
  Problem: We have the entire Arithmetic Hierarchy as soon as we have higher order quantifiers (using the standard approach).

Programme:

1. Develop a more finely stratified hierarchy.
2. Define device-independent as "below Σ_3^0"
3. Search for upper and lower bounds on $\mathcal{C}$ in this hierarchy.