Synthetic Versions of the Kleene-Post and Post’s Theorem

Dominik Kirst\textsuperscript{1}, Niklas Mück\textsuperscript{1}, and Yannick Forster\textsuperscript{1,2}

\textsuperscript{1} Saarland University, Saarland Informatics Campus
\textsuperscript{2} Inria, Gallinette Project-Team, Nantes, France

Abstract

We discuss our ongoing formalisation of the Kleene-Post theorem ([6], establishing incomparable Turing degrees) and Post’s theorem ([9], connecting the arithmetic hierarchy with Turing degrees) using synthetic computability theory in constructive type theory.

Synthetic Oracle Machines We briefly outline the synthetic rendering of oracle machines as described in a related abstract [4], adjusting [2], based on a similar proposal by Bauer [1]. The main idea is that an oracle machine $R$ can be represented as a function operating on functional relations $A : \mathbb{N} \rightarrow \mathbb{B}$ relating (some) natural numbers $n : \mathbb{N}$ to (unique) boolean values $b : \mathbb{B}$:

$$R : (\mathbb{N} \rightarrow \mathbb{B}) \rightarrow \mathbb{N} \rightarrow \mathbb{B}$$

The input relation acts as oracle that can be accessed to describe the returned relation. To ensure that this description is effective, we require $R$ to return computable output for computable input, captured as partial functions $f : \mathbb{N} \rightarrow \mathbb{B}$, by imposing a computational core $r : (\mathbb{N} \rightarrow \mathbb{B}) \rightarrow \mathbb{N} \rightarrow \mathbb{B}$ with

$$\forall f. R f = r f.$$ 

Note that here and in the remainder of this text we freely identify partial functions $f : \mathbb{N} \rightarrow \mathbb{B}$ with their (functional) graphs $\lambda n b. f n = b$, reusing the equality symbol for evaluation of $f$. To further rule out exotic behaviour, we require $R$ to be continuous in the following sense:

$$\forall(A : \mathbb{N} \rightarrow \mathbb{B}), \forall(n : \mathbb{N}), \forall(b : \mathbb{B}). \forall \sigma, \tau. R A n b \rightarrow \exists(L : \mathbb{N}^*). L \subseteq \text{dom}(A) \land \forall \sigma'. A' =_L A \rightarrow R A' n b$$

Continuity in this sense expresses that from any terminating run $R A n b$ one can extract a list $L$ of queries to which the oracle $A$ replied, such that $RA'n$ terminates for all oracles $A'$ agreeing with $A$ on $L$ with the same value $b$. Observed externally, by all these restrictions and according to the classical, syntactic definition we narrow down the amount of oracle machines to countable extent. In fact, we make this limitation available internally by assuming an enumeration $r_n$ of all computational cores. We currently investigate for which formulations a variant of Church’s thesis [7, 10, 11, 3] is enough to obtain such an enumerator.

Given $A, B : \mathbb{N} \rightarrow \mathbb{P}$, we call $R$ a Turing reduction from $A$ to $B$ if $RB = A$ (reinterpreting predicates as functional relations) and write $A \preceq_T B$ if a Turing reduction from $A$ to $B$ exists. We assume extensionality of functions and relations.

Kleene-Post Theorem To establish incomparable Turing degrees, we adapt the proof given in Odifreddi’s textbook [8] to our synthetic setting. The usual strategy is to obtain them as the unions $A := \bigcup_{n \in \mathbb{N}} \sigma_n$ and $B := \bigcup_{n \in \mathbb{N}} \tau_n$ of cumulative increasing sequences $\sigma_n$ and $\tau_n$ of boolean strings such that the former take care that no $r_n$ induces a reduction $B \preceq_T A$ and the latter conversely rule out $A \preceq_T B$. Naturally, in our synthetic setting we are not able to define these sequences as computable functions $\mathbb{N} \rightarrow \mathbb{B}^*$, as this would force $A$ and $B$ decidable. Instead, we characterise both sequences simultaneously with an inductive predicate $\triangleright : \mathbb{N} \rightarrow \mathbb{B}^* \rightarrow \mathbb{B}^* \rightarrow \mathbb{P}$ such that $n \triangleright (\sigma, \tau)$ represents $\sigma_n$, as $\sigma$ and $\tau_n$ as $\tau$, by adding to $0 \triangleright (\epsilon, \epsilon)$ the rules:

$$\frac{2n \triangleright (\sigma, \tau)}{2n + 1 \triangleright (\sigma', \tau + [\neg \theta])} \quad \frac{2n + 1 \triangleright (\sigma, \tau)}{2n + 2 \triangleright (\sigma + [\neg \theta], \tau')} \quad \frac{2n + 1 \triangleright (\sigma, \tau)}{2n + 1 \triangleright (\sigma, \tau + [\theta])} \quad \frac{2n + 1 \triangleright (\sigma, \tau)}{2n + 2 \triangleright (\sigma + [\theta], \tau')}$$

$$\frac{2n \triangleright (\sigma, \tau)}{2n + 2 \triangleright (\sigma' \geq \sigma \land b = r_n \sigma' [\tau])} \quad \frac{2n + 1 \triangleright (\sigma, \tau)}{2n + 2 \triangleright (\sigma + [\theta], \tau')}$$

$$\frac{2n \triangleright (\sigma, \tau)}{2n + 1 \triangleright (\sigma', \tau + [\neg \theta])} \quad \frac{2n + 1 \triangleright (\sigma, \tau)}{2n + 2 \triangleright (\sigma' \geq \sigma \land b = r_n \sigma' [\tau])} \quad \frac{2n + 1 \triangleright (\sigma, \tau)}{2n + 2 \triangleright (\sigma + [\theta], \tau')}$$

$$\frac{2n \triangleright (\sigma, \tau)}{2n + 2 \triangleright (\sigma' \geq \sigma \land b = r_n \sigma' [\tau])} \quad \frac{2n + 1 \triangleright (\sigma, \tau)}{2n + 2 \triangleright (\sigma + [\theta], \tau')}$$

$$\frac{2n \triangleright (\sigma, \tau)}{2n + 2 \triangleright (\sigma' \geq \sigma \land b = r_n \sigma' [\tau])} \quad \frac{2n + 1 \triangleright (\sigma, \tau)}{2n + 2 \triangleright (\sigma + [\theta], \tau')}$$

$\triangleright$
In every even step with \(2n \triangleright (\sigma, \tau)\) the sequences are extended such that \(r_n\) applied to any prefix of \(A\) differs from any prefix of \(B\) at position \(|\tau|\), either by flipping the result if \(r_n\) already converges on some extension \(\sigma' \geq \sigma\) (E1) or by setting a dummy value if \(r_n\) diverges on all extensions (E2). Dually, in every odd step with \(2n+1 \triangleright (\sigma', \tau')\) it is taken care that \(r_e\) applied to any prefix of \(B\) differs from any prefix of \(A\).

We state the central lemma used to show \(B \not\leq_T A\), a dual version yields \(A \not\leq_T B\).

**Lemma 1.** Let \(R\) be an oracle machine factoring through the computational core \(r_n\). If further given \(2n \triangleright (\sigma, \tau)\) and \(2n + 1 \triangleright (\sigma', \tau')\), then \(B |\tau| b\) implies \(\neg (RA |\tau| b)\).

**Theorem 1** (Kleene-Post). There are predicates \(A, B\) such that neither \(A \preceq_T B\) nor \(B \preceq_T A\).

**Proof.** Suppose that \(B \preceq_T A\), so \(RA = B\) for some oracle machine \(R\) with core \(r_n\). Given that we try to derive a contradiction, we can argue classically enough to obtain \(2n \triangleright (\sigma, \tau), 2n+1 \triangleright (\sigma', \tau'),\) and \(B |\tau| b\). Then by Lemma 1 we obtain \(\neg (RA |\tau| b)\), contradicting \(RA = B\). \(\square\)

**Post’s Theorem** To connect the arithmetical hierarchy with the structure of Turing degrees, we again follow a usual textbook presentation translated to constructive type theory. To be able to state the theorem in our setting, we render all involved notions synthetically.

First, we represent the arithmetical hierarchy with a mutually inductive predicate:

\[
\begin{align*}
\Sigma_0^k (\lambda \vec{n}. f \vec{n} = \mathrm{true}) & \quad \Pi_0^k (\lambda \vec{n}. f \vec{n} = \mathrm{true}) & \quad \Pi_{n+1}^k p & \quad \Sigma_{n+1}^k p \\
\Sigma_0^k (\lambda \vec{n}. \exists x. p (x :: \vec{n})) & \quad \Pi_{n+1}^k (\lambda \vec{n}. \forall x. p (x :: \vec{n}))
\end{align*}
\]

The first two rules assert that \(k\)-ary decidable predicates form the base of the hierarchy. The third rule states that for a \(\Pi_n\) predicate \(p\) of arity \(k+1\) the \(k\)-ary predicate obtained by capturing the first variable of \(p\) by an existential quantifier is \(\Sigma_{n+1}\). The fourth rule dually expresses how a \(\Sigma_n\) predicate is turned into \(\Pi_{n+1}\) with a universal quantifier. As a sanity check, using a form of Church’s thesis for a concrete model of computation, we can show the equivalence of our synthetic characterisation of the arithmetical hierarchy with a more conventional definition using first-order formulas in the language of arithmetic, as mechanism in [5].

Secondly, we define the Turing jump \(A'\) of a predicate \(A\) using the core enumeration \(r_n\):

\[A' := \lambda n. \exists R. \forall f. R f = r_n f \land RA n \text{ true}\]

This definition expresses the self-halting problem for oracle machines as it contains exactly those numbers \(n\) such that the \(n\)-th oracle machine \(R\) (as characterised by \(r_n\)) used with an oracle for \(A\) accepts \(n\). We denote the \(n\)-th Turing jump of the empty predicate by \(\emptyset^{(n)}\).

Finally, we say that \(A\) is semi-decidable relative to \(B\) if there is an oracle machine \(R\) with

\[\forall n. A n \leftrightarrow R B n \text{ true}\]

The hardest part of Post’s theorem is to show that \(RA = \Sigma_1\) relative to \(A\) by showing:

**Lemma 2.** Given an oracle machine \(R\) with core \(r\), termination \(RA n b\) is equivalent to

\[\exists sol. (\forall n \in sol. \text{true} \land A b \text{true}) \land (\forall n \in sol. \text{false} \land A b \text{false}) \land R \text{ lookup sol sol. true} n = b\]

where \(\text{lookup sol sol. true sol. true} n\) returns \text{true} if \(n \in sol\), \text{false} if \(n \in sol\), and diverges otherwise.

We conclude Post’s theorem in a common formulation, employing our synthetic definitions.

**Theorem 2** (Post). Assuming LEM (\(\forall p. p \lor \neg p\)), the following can be shown:

- A unary predicate \(A\) is \(\Sigma_{n+1}\) iff it is semi-decidable relative to \(\emptyset^{(n)}\).
- If \(A = \Sigma_n\), then \(A \preceq_T \emptyset^{(n)}\). If \(n > 0\) already \(A \preceq_m \emptyset^{(n)}\) for synthetic many-one reductions.

In our current mechanisation, we assume LEM to allow switching between \(\Sigma_n\) and \(\Pi_n\) by complementation. We currently investigate how this assumption can be weakened or eliminated.
References


