

# Randomising Realisability<sup>\*</sup>

Merlin Carl<sup>1</sup>, Lorenzo Galeotti<sup>2</sup>, and Robert Passmann<sup>3,4</sup>

<sup>1</sup> Europa-Universität Flensburg, 24943 Flensburg, Germany

<sup>2</sup> Amsterdam University College, Postbus 94160, 1090 GD Amsterdam, The Netherlands

<sup>3</sup> Institute for Logic, Language and Computation, Faculty of Science, University of Amsterdam, P.O. Box 94242, 1090 GE Amsterdam, The Netherlands

<sup>4</sup> St John's College, University of Cambridge, Cambridge CB2 1TP, England

**Abstract.** We consider a randomised version of Kleene's realisability interpretation of intuitionistic arithmetic in which computability is replaced with randomised computability with positive probability. In particular, we show that (i) the set of randomly realisable statements is closed under intuitionistic first-order logic, but (ii) different from the set of realisable statements, that (iii) "realisability with probability 1" is the same as realisability and (iv) that the axioms of bounded Heyting's arithmetic are randomly realisable, but some instances of the full induction scheme fail to be randomly realisable.

## 1 Introduction

Have you met skeptical Steve? Being even more skeptical than most mathematicians, he only believes what he actually sees. To convince him that there is an  $x$  such that  $A$ , you have to give him an example, together with evidence that  $A$  holds for that example. To convince him that  $A \rightarrow B$ , you have to show him a *method* for turning evidence of  $A$  into evidence of  $B$ , and so on. Given that Steve is "a man provided with paper, pencil, and rubber, and subject to strict discipline" [9], we can read "method" as "Turing program", which leads us to Kleene's realisability interpretation of intuitionistic logic [4].

Steve has a younger brother, pragmatical Per. Like Steve, Per is equipped with paper and pencil; however, he also has a coin on his desk, which he is allowed to throw from time to time while performing computations. By his pragmatical nature, he does not require being successful at obtaining evidence for a given proposition  $A$  every time he gives it a try; he is quite happy when it works with probability  $(1 - \frac{1}{10^{100}})$  or so, which makes it highly unlikely to ever fail in his lifetime.

Per wonders whether his pragmatism is more powerful than Steve's method. After all, he knows about Sacks's theorem [1, Corollary 8.12.2] that every function  $f : \omega \rightarrow \omega$  that is computable using coin throws with positive probability is recursive. Can he find evidence for some claims where Steve fails? He also notices

---

<sup>\*</sup> The authors would like to thank Rosalie Iemhoff and Jaap van Oosten for discussions about the material included in this paper.

that turning such “probabilistic evidence” for  $A$  into “probabilistic evidence” for  $B$  is a job considerably different (and potentially harder) than turning evidence for  $A$  into evidence for  $B$ . Could it be that there are propositions whose truth Steve can see, but Per cannot? Although Per is skeptical, e.g., of the law of the excluded middle just like Steve, he is quite fond of the deduction rules of intuitionistic logic; thus, he wonders whether the set of statements for which he can obtain his “highly probably evidence” is closed under these.

Steve is unhappy with his brother’s sloppiness. After all, even probability  $(1 - \frac{1}{10^{100}})$  leaves a small, albeit nonzero, chance of getting things wrong. He might consider changing his mind if that chance was brought down to 0 by strengthening Steve’s definition, demanding that the “probabilistic evidence” works with probability 1. However, he is only willing to give up absolute security if that leads to evidence for more statements. Thus, he asks whether “probability 1 evidence” is the same as “evidence”.

These and other questions will be considered in this paper. To begin with, we will model Per’s attitude formally, which gives us the concepts of  $\mu$ -realisability and *almost sure realisability*. We will then show the following: There are statements that are  $\mu$ -realisable, but not realisable (Theorem 14). The set of  $\mu$ -realisable statements are closed under deduction in intuitionistic predicate calculus (Theorem 18); in a certain sense to be specified below, the law of excluded middle fails for  $\mu$ -realisability (Lemma 16). The axioms of Heyting arithmetic except for the induction schema are  $\mu$ -realised (Theorem 19); and there are instances of the induction schema that are not  $\mu$ -realised (Theorem 20). Almost sure realisability is the same as realisability (Theorem 25).

## 2 Preliminaries

Realisability is one of the most common semantic tools for the study of constructive theories and was introduced by Kleene in his seminal 1945 paper [4]. In this work, Kleene connected intuitionistic arithmetic—nowadays called *Heyting arithmetic*—and recursive functions. The essential idea is that a statement is true if and only if there is a recursive function witnessing its truth. For more details on realisability, see also Troelstra’s 344 [8], and van Oosten’s paper [7] for an excellent historical survey of realisability. In particular, see [8, Definition 3.2.2] for a definition of realisability in terms of recursive functions. In what follows, we denote this classical relation of realisability by ‘ $\Vdash$ ’.

As mentioned in the introduction, we want to give pragmatic Per the ability to throw coins while he tries to prove the truth of a statement. We will implement this coin throwing by allowing Per to access an infinite binary sequence. Therefore, we will make use of the Lebesgue measure on Cantor space  $2^\omega$ . For a full definition, see Kanamori’s section on ‘Measure and Category’ [2, Chapter 0]. We denote the Lebesgue measure by  $\mu$ . Recall that a set  $A$  is Lebesgue measurable if and only if there is a Borel set  $B$  such that the symmetric difference of  $A$  and  $B$  is null. Given an element  $u$  of Cantor space we will denote by  $N_{u \upharpoonright n}$  the basic clopen set  $\{v \in 2^\omega ; u \upharpoonright n \subset v\}$  where as usual  $u \upharpoonright n$  is the prefix of  $u$

of length  $n$ , and  $u \upharpoonright n \subset v$  if  $u \upharpoonright n$  is a prefix of  $v$ . We recall that given a binary sequence of length  $n$ , we have that  $N_s$  is measurable and  $\mu(N_s) = \frac{1}{2^n}$ .

We fix a computable enumeration  $(p_n)_{n \in \mathbb{N}}$  of programs. Moreover, given a program  $p$  that uses an oracle and an element  $u \in 2^\omega$  we will denote by  $p^u$  the program  $p$  where the oracle tape contains  $u$  at the beginning of the computation. Moreover, given  $n \in \mathbb{N}$  we will denote by  $p(n)$  the program that for every oracle  $u \in 2^\omega$  returns  $p^u(n)$ .

A sentence in the language of arithmetic is said to be  $\Delta_0$  if it does not contain unbounded quantifiers. We will say that a sentence is a *pretty*  $\Sigma_1$  if it is  $\Delta_0$  or of the form  $Q_0 Q_1 \dots Q_n \psi$  where  $\psi$  is  $\Delta_0$  and  $Q_i$  is either an existential quantifier or a bounded universal quantifier for every  $0 \leq i \leq n$ . Similarly, we will say that a sentence is a *universal*  $\Pi_1$  if it is  $\Delta_0$  or of the form  $Q_0 Q_1 \dots Q_n \psi$  where  $\psi$  is  $\Delta_0$  and  $Q_i$  is a universal quantifier for every  $0 \leq i \leq n$ .

Throughout this paper, we fix codings for formulas and programs. In order to simplify notation, we will use  $\varphi$  to refer to both the formula and its code, and similar for programs  $p$ . We end this section with some lemmas on realisability of pretty  $\Sigma_1$  and universal  $\Pi_1$  formulas.

**Lemma 1.** *There is a program  $p$  that for every pretty  $\Sigma_1$  sentence  $\varphi$  does the following: If  $\varphi$  is true then  $p(\varphi)$  halts and outputs a realiser of  $\varphi$  and otherwise, it diverges.*

*Proof.* First we define the program for  $\Delta_0$  formulas by recursion.

(1) If  $\varphi$  is atomic,  $p$  first checks if  $\varphi$  is true. If so then  $p$  returns any natural number, otherwise, it loops.

(2)  $\varphi \equiv \psi_0 \wedge \psi_1$ : the program  $p$  checks whether  $\psi_0$  and  $\psi_1$  are true. If both computations are successful then  $p$  returns the code of a program  $q$  which returns  $p(\psi_0)$  on input 0 and  $p(\psi_1)$  on input 1.

(3)  $\varphi \equiv \psi_0 \vee \psi_1$  the program  $p$  starts checking if at least one between  $\psi_0$  and  $\psi_1$  is true. If one of the two computations is successful then  $p$  returns the code of a program  $q$  which returns  $p(\psi_i)$  on input 1 and  $i$  on input 0 where  $i$  is the smallest  $i$  such that  $\psi_i$  is true. Otherwise the program loops.

(4)  $\varphi \equiv \psi_0 \rightarrow \psi_1$  then  $p$  first checks whether  $\psi$  is true if not  $p$  returns 0 otherwise returns a program that for every input returns  $p(\psi_1)$ .

(5)  $\varphi \equiv \exists x < n \psi$  then the program  $p$  checks if there is  $m < n$  such that  $\psi(m)$  is true. If so then  $p$  returns the code of a program  $q$  which returns  $p(\psi(m))$  on input 0 and  $m$  on input 1 where  $m$  is the smallest natural number such that  $\psi(m)$  is true. Otherwise the program loops.

(6)  $\varphi \equiv \forall x < n \psi$  then  $p$  checks in parallel the truth of all the instances of  $\psi(m)$  for  $m < n$ . If all of them are true then  $p$  returns the code of a program  $q$  which for all  $m \in \mathbb{N}$  returns  $p(\psi(m))$ . Otherwise the program loops.

Now we extend the definition of  $p$  to pretty  $\Sigma_1$  sentences. Assume that  $\varphi$  is of the form  $Q_1 \dots Q_n \psi$  where  $\psi$  is  $\Delta_0$  and  $Q_i$  is either an existential quantifier or a bounded universal quantifier for every  $0 \leq i \leq n$ . We define  $p$  by recursion on  $n$ . Since the base case and the inductive step are essentially the same we will only show the latter.

Let  $n = m + 1$  and  $\varphi \equiv Q_0 Q_1 \dots Q_n \psi$  where  $\psi$  is  $\Delta_0$ . We assume that  $f$  is already defined for  $Q_1 \dots Q_n \psi$  and need to show that we can extend it to  $\varphi$ . We have two cases

(1)  $Q_0$  is a bounded quantifier. Then we repeat what we did in part (5) and (6) of this proof.

(2)  $Q_0$  is an unbounded existential quantifier. Then the program  $p$  starts an unbounded search to find an  $i$  such that  $\psi(i)$  is true. If it finds it then  $p$  returns the code of a program  $q$  which returns  $p(\psi(i))$  on input 0 and  $i$  on input 1 where  $i$  is the smallest natural number such that  $\psi(i)$  is true.

**Lemma 2.** *There is a program  $p$  that for every universal  $\Pi_1$  sentence  $\varphi$  does the following: If  $\varphi$  is true then  $p(\varphi)$  halts and outputs a realiser of  $\varphi$  (we do not specify a behaviour otherwise).*

*Proof.* Define  $p$  as in the proof of Lemma 1 for  $\Delta_0$  formulas. Then for formulas of the type  $\forall x \psi$  let  $p(\psi)$  be the code of the program that for all  $n$  runs  $p(\psi(n))$ .

**Lemma 3.** *A pretty  $\Sigma_1$  sentence in the language of arithmetic is realised if and only if it is true. The same result holds for universal  $\Pi_1$  sentences.*

*Proof.* The right-to-left direction follows from Lemma 1. The other direction is a straightforward induction on the complexity of  $\varphi$ . The proof for universal  $\Pi_1$  sentences is also an easy induction.

**Corollary 4.** *There is a program  $p$  that for every pretty  $\Sigma_1$  sentence  $\varphi$  does the following: If  $\varphi$  is true then  $p(\varphi)$  halts and outputs a realiser of  $\varphi$  and otherwise, it diverges.*

### 3 Random Realisability

In this section we will introduce the notion of  $\mu$ -realisability and prove the basic properties of this relation. As we mentioned before, we will modify classical realisability in order to use realisers that can access an element of Cantor space. Then we will say that a sentence is randomly realised if for non-null set of oracles in Cantor space the program does realise the sentence. Formally we define  $\mu$ -realisability as follows:

**Definition 5 ( $\mu$ -Realisability).** *We define two relations  $\Vdash_{\mathcal{O}}$  and  $\Vdash_{\mu}$  by mutual recursion. Let  $u \in 2^{\omega}$ ,  $p$  be a program that uses an oracle, and  $\varphi$  be a sentence in the language of arithmetic. We define:*

1.  $(p, u), \not\vdash_{\mathcal{O}} \perp$ ,
2.  $(p, u) \Vdash_{\mathcal{O}} n = m$  iff  $n = m$ ,
3.  $(p, u) \Vdash_{\mathcal{O}} \varphi \wedge \psi$  iff  $(p^u(0), u) \Vdash_{\mathcal{O}} \varphi$  and  $(p^u(1), u) \Vdash_{\mathcal{O}} \psi$ ,
4.  $(p, u) \Vdash_{\mathcal{O}} \varphi \vee \psi$  iff we have  $p^u(0) = 0$  and  $(p^u(1), u) \Vdash_{\mathcal{O}} \varphi$  or  $p^u(0) = 1$  and  $(p^u(1), u) \Vdash_{\mathcal{O}} \psi$ ,
5.  $(p, u) \Vdash_{\mathcal{O}} \varphi \rightarrow \psi$  iff for all  $s$  such that  $s \Vdash_{\mu} \varphi$ , we have that  $p^u(s) \Vdash_{\mu} \psi$ ,
6.  $(p, u) \Vdash_{\mathcal{O}} \exists \mathbf{x} \varphi$  iff  $(p^u(0), u) \Vdash_{\mathcal{O}} \varphi(p^u(1))$ ,
7.  $(p, u) \Vdash_{\mathcal{O}} \forall \mathbf{x} \varphi$  iff for all  $n \in \omega$  we have  $(p^u(n), u) \Vdash_{\mathcal{O}} \varphi(n)$ ,

For every program  $p$  that uses an oracle and every sentence  $\varphi$  in the language of arithmetic, we will denote by  $C_{p,\varphi}$  the set:  $\{u \in 2^\omega ; (p, u) \Vdash_{\mathcal{O}} \varphi\}$ . Let  $\varphi$  be a sentence in the language of arithmetic,  $r$  be a positive real number, and  $p$  be a natural number. We define  $p \Vdash_{\mu} \varphi \geq r$  as follows:  $p \Vdash_{\mu} \varphi \geq r$  iff  $\mu(C_{p,\varphi}) \geq r$ .

In this case we will say that  $p$  randomly realises (or  $\mu$ -realises)  $\varphi$  with probability at least  $r$ . We will say that  $\varphi$  is randomly realisable (or  $\mu$ -realisable) with probability at least  $r$  if and only if there is  $p$  such that  $p \Vdash_{\mu} \varphi \geq r$ . Moreover, we write  $p \Vdash_{\mu} \varphi$  and say that  $p$  randomly realises (or  $\mu$ -realises)  $\varphi$  if and only if  $p \Vdash_{\mu} \varphi \geq r$  for some  $r > 0$ . Finally, we will say that  $\varphi$  is randomly realisable (or  $\mu$ -realisable) if  $\sup\{\mu(C_{p,\varphi}) ; p \Vdash_{\mu} \varphi\} = 1$ .

Why is it not possible to give a simpler definition of  $\Vdash_{\mu}$ ? A natural attempt would be the following:  $p \Vdash_{\mu} \varphi \geq r \Leftrightarrow \mu(\{u ; (p, u) \Vdash_{\mathcal{O}_r} \varphi\}) \geq r$  where  $\Vdash_{\mathcal{O}_r}$  denotes oracle-realisability, obtained by replacing computability with computability relative to a fixed oracle in Kleene realisability. Unfortunately, it turns out that this relation is not closed under modus ponens and the  $\forall$ -GEN rule of predicate logic. Another natural approach is the one of §6. We start our study of  $\mu$ -realisability by showing that the set of  $\mu$ -realised sentences of arithmetic is consistent.

**Lemma 6.** *Let  $\varphi$  a sentence in the language of arithmetic. Then  $\varphi$  is  $\mu$ -realised iff  $\neg\varphi$  is not  $\mu$ -realised.*

*Proof.* Assume that both  $p \Vdash_{\mu} \varphi$  and  $q \Vdash_{\mu} \neg\varphi$ . Then for all  $u \in C_{q,\neg\varphi}$  we have that  $q^u(p)$  would be a realiser of  $\perp$ . But this is a contradiction.

The following lemma has a crucial role in the theory of  $\mu$ -realisability.

**Lemma 7 (Push Up Lemma).** *Let  $\varphi$  be a sentence in the language of first order arithmetic and  $0 < r \leq r' < 1$  be positive real numbers. Then  $\varphi$  is randomly realisable with probability at least  $r$  if and only if  $\varphi$  is randomly realisable with probability at least  $r'$ .*

*Proof.* The right-to-left direction is trivial. For the left-to-right direction, let  $\varphi$  be randomly realisable with probability at least  $r$ . We will show that  $\varphi$  is randomly realisable with probability at least  $r'$ . Let  $p$  be a program such that  $\mu(C_{p,\varphi}) \geq r > 0$ . By the Lebesgue Density Theorem [3, Exercise 17.9] there are  $u \in 2^\omega$  and  $n \in \omega$  such that  $\frac{\mu(C_{p,\varphi} \cap N_{u \upharpoonright n})}{\mu(N_{u \upharpoonright n})} > r'$ . Now, let  $p'$  be the program that given an oracle runs  $p$  with oracle  $(u \upharpoonright n) \circ u$ . Note that  $\mu(C_{p',\varphi}) = \frac{\mu(C_{p,\varphi} \cap N_{u \upharpoonright n})}{\mu(N_{u \upharpoonright n})} > r'$ . Finally,

it follows trivially by the definition that  $p'$  randomly realisable with probability at least  $r'$  as desired.

By the Push Up Lemma, we can simplify our definition of  $\mu$ -realisability.

**Corollary 8.** *A sentence  $\varphi$  in the language of arithmetic is  $\mu$ -realisable if and only if there are  $0 < r \in \mathbb{R}$  and  $p$  such that that  $\mu$ -realises  $\varphi$  with probability at least  $r$ .*

We conclude this section by proving some basic interactions between  $\mu$ -realisability and the logical operators.

**Lemma 9.** *For all programs  $p$  and sentences  $\varphi$  the set  $C_{p,\varphi}$  is Borel. In particular  $C_{p,\varphi}$  is measurable.*

*Proof.* The proof is an induction on the complexity of  $\varphi$ . All the cases except implication follow directly from the closure properties of the pointclass of Borel sets, see, e.g., [5, Theorem 1C.2]. Let us just prove the implication case. Let  $\varphi \equiv \psi_0 \rightarrow \psi_1$  and  $p$  be a program. For every program  $s$  let  $A_s$  be  $2^\omega$  if  $s \Vdash_\mu \psi_0$  and  $C_{p(s),\psi_1}$ , otherwise. Then  $C_{p,\varphi} = \bigcap_{s \in \mathbb{N}} A_s$ . By inductive hypothesis we have that  $A_s$  is Borel for every  $s$  so  $C_{p,\varphi}$  is a countable intersection of Borel sets, which is Borel.

**Corollary 10.** *Let  $\psi_0$  and  $\psi_1$  be sentences and let  $\varphi$  be a formula. Then for every  $p$  the following hold:*

1.  $p \Vdash_\mu \psi_0 \wedge \psi_1$  if and only if there are  $s$  and  $q$  such that  $s \Vdash_\mu \psi_0$  and  $q \Vdash_\mu \psi_1$ .
2.  $p \Vdash_\mu \psi_0 \vee \psi_1$  if and only if there is  $q$  such that  $q \Vdash_\mu \psi_0$  or  $q \Vdash_\mu \psi_1$ .
3. If  $p \Vdash_\mu \psi_0 \rightarrow \psi_1$  then  $p(s) \Vdash_\mu \psi_1$  for all  $s$  such that  $s \Vdash_\mu \psi_0$ .
4. If  $p \Vdash_\mu \exists x \varphi$  then there there is  $n \in \mathbb{N}$  such that  $p(0) \Vdash_\mu \varphi(n)$ .
5. If  $p \Vdash_\mu \forall x \varphi$  then for all  $n \in \mathbb{N}$  we have  $p(n) \Vdash_\mu \varphi(n)$ .

*Proof.* Note that an case-by-case proof shows that for every  $n \in \mathbb{N}$ ,  $u \in 2^\omega$ , and formula  $\varphi$  we have that:

$$(p(n), u) \Vdash_{\circ} \varphi \text{ iff } (p^u(n), u) \Vdash_{\circ} \varphi.$$

(1) First assume that  $p \Vdash_\mu \psi_0 \wedge \psi_1 \geq r$ . Then we have that  $\mu(C_{p,\psi_0 \wedge \psi_1}) \geq r$  and for all  $u \in C_{p,\psi_0 \wedge \psi_1}$  we have that  $(p, u) \Vdash_{\circ} \psi_0 \wedge \psi_1$ . But then for all  $u \in C_{p,\psi_0 \wedge \psi_1}$  we have that  $(p^u(0), u) \Vdash_{\circ} \psi_0$  and  $(p^u(0), u) \Vdash_{\circ} \psi_1$ . So for every  $u \in C_{p,\psi_0 \wedge \psi_1}$  we have that  $(p(0), u) \Vdash_{\circ} \psi_0$  and  $(p(1), u) \Vdash_{\circ} \psi_1$  but then  $p(0) \Vdash_\mu \psi_0 \geq r$  and  $p(1) \Vdash_\mu \psi_1 \geq r$ . Let  $s = p(0)$  and  $q = p(1)$ .

Now assume that  $q \Vdash_\mu \psi_0$  and  $s \Vdash_\mu \psi_1$ . Therefore for all  $u \in C_{q,\psi_0}$  and  $v \in C_{s,\psi_1}$  we have that  $(q, u) \Vdash_{\circ} \psi_0$  and  $(s, v) \Vdash_{\circ} \psi_1$ . By Lemma 7 we can assume that  $s$  and  $q$  are such that  $\mu(C_{q,\psi_0} \cap C_{s,\psi_1}) > r$ . But then if we let  $p$  be the program that returns  $q$  on input 0 and  $s$  on input 1, we have that for all  $u \in C_{q,\psi_0} \cap C_{s,\psi_1}$  we have that  $(p(0), u) \Vdash_{\circ} \psi_0$  and  $(p(1), u) \Vdash_{\circ} \psi_1$ . Therefore  $p \Vdash_\mu \psi_0 \wedge \psi_1$  as desired.

(2) First assume that  $p \Vdash_{\mu} \psi_0 \vee \psi_1$ . Then we have that  $\mu(C_{p, \psi_0 \vee \psi_1}) > 0$  and for all  $u \in C_{p, \psi_0 \vee \psi_1}$  we have that  $(p, u) \Vdash_{\circ} \psi_0 \vee \psi_1$ . But then for all  $u \in C_{p, \psi_0 \vee \psi_1}$  we have that  $(p^u(1), u) \Vdash_{\circ} \psi_0$  or  $(p^u(1), u) \Vdash_{\circ} \psi_1$ . Moreover, note that by Lemma 9 we have that both  $C_{p(1), \psi_0}$  and  $C_{p(1), \psi_1}$  are measurable sets and since  $C_{p, \psi_0 \vee \psi_1} \subseteq C_{p(1), \psi_0} \cup C_{p(1), \psi_1}$  at least one of them is not null. Without loss of generality assume that  $\mu(C_{p(1), \psi_0}) > 0$ . So,  $(p(1), u) \Vdash_{\circ} \psi_0$  for all  $u \in C_{p(1), \psi_0}$ . Let  $q$  be the program that for every oracle returns  $p^u(1)$  on input 1 and 0 on input 0. Then trivially  $q \Vdash_{\mu} \psi_0$  as desired.

Now assume that  $q \Vdash_{\mu} \psi_0$ , the same proof works in the case in which  $q \Vdash_{\mu} \psi_1$ . Therefore for all  $u \in C_{q, \psi_0}$  we have that  $(q, u) \Vdash_{\circ} \psi_0$ . Then if we let  $p$  be the program that returns  $q$  on input 1 and 0 on input 0, we have that for all  $u \in C_{q, \psi_0}$   $(p(1), u) \Vdash_{\circ} \psi_0$ . Therefore  $p \Vdash_{\mu} \psi_0 \vee \psi_1$  as desired.

(3) Assume that  $p \Vdash_{\mu} \psi_0 \rightarrow \psi_1$ . Then we have that  $\mu(C_{p, \psi_0 \rightarrow \psi_1}) > 0$  and for all  $u \in C_{p, \psi_0 \rightarrow \psi_1}$  we have that  $(p, u) \Vdash_{\circ} \psi_0 \rightarrow \psi_1$ . But then for all  $u \in C_{p, \psi_0 \rightarrow \psi_1}$  and for every  $s$  such that  $s \Vdash_{\mu} \psi_0$  we have  $p^u(s) \Vdash_{\mu} \psi_1$ . But for all  $s$  such that  $s \Vdash_{\mu} \psi_0$  we have that  $(p^u(s), u) \Vdash_{\circ} \psi_1$  and therefore  $(p(s), u) \Vdash_{\circ} \psi_1$ . So,  $p(s) \Vdash_{\mu} \psi_1$  for all  $s$  such that  $s \Vdash_{\mu} \psi_0$  as desired.

(4) Assume that  $p \Vdash_{\mu} \exists x \psi$ . Then we have that  $\mu(C_{p, \exists x \psi}) > 0$  and for all  $u \in C_{p, \exists x \psi}$  we have that  $(p, u) \Vdash_{\circ} \exists x \psi$ . But then for all  $u \in C_{p, \exists x \psi}$  we have that  $(p^u(0), u) \Vdash_{\circ} \psi(p^u(1))$ . Moreover, note that by Lemma 9 we have that for every  $n \in \mathbb{N}$  the set  $C_{p(0), \psi(n)}$  is measurable and since  $C_{p, \exists x \psi} \subseteq \bigcup_{n \in \mathbb{N}} C_{p(0), \psi(n)}$  there is  $n \in \mathbb{N}$  such that  $\mu(C_{p(0), \psi(n)}) > 0$ . So,  $(p(0), u) \Vdash_{\circ} \psi(n)$  for all  $u \in C_{p(0), \psi(n)}$  and therefore  $p(0) \Vdash_{\mu} \psi(n)$  as desired.

(4) Assume that  $p \Vdash_{\mu} \forall x \psi$ . Then we have that  $\mu(C_{p, \forall x \psi}) > 0$  and for all  $u \in C_{p, \forall x \psi}$  we have that  $(p, u) \Vdash_{\circ} \forall x \psi$ . But then for all  $u \in C_{p, \forall x \psi}$  and every  $n \in \mathbb{N}$  we have that  $(p^u(n), u) \Vdash_{\circ} \psi(n)$ . So,  $(p(n), u) \Vdash_{\circ} \psi(n)$  and  $p(n) \Vdash_{\mu} \psi(n)$  for all  $n \in \mathbb{N}$ .

**Corollary 11.** *Let  $\psi_0$  and  $\psi_1$  be sentences and let  $\varphi$  be a formula. Then for every  $p$  the following hold:*

1.  $p \Vdash_{\mu} \psi_0 \wedge \psi_1 \geq 1$  if and only if  $p(0) \Vdash_{\mu} \psi_0 \geq 1$  and  $p(1) \Vdash_{\mu} \psi_1 \geq 1$ .
2. If  $p(1) \Vdash_{\mu} \psi_0 \geq 1$  or  $p(1) \Vdash_{\mu} \psi_1 \geq 1$  then  $p \Vdash_{\mu} \psi_0 \vee \psi_1 \geq 1$ .
3. If  $p(s) \Vdash_{\mu} \psi_1 \geq 1$  for all  $s$  such that  $s \Vdash_{\mu} \psi_0$  then  $p \Vdash_{\mu} \psi_0 \rightarrow \psi_1 \geq 1$ .
4. If there is  $n \in \mathbb{N}$  such that  $p(n) \Vdash_{\mu} \varphi(n) \geq 1$  then  $p \Vdash_{\mu} \exists x \varphi \geq 1$ .
5. If all  $n \in \mathbb{N}$  we have  $p(n) \Vdash_{\mu} \varphi(n) \geq 1$  then  $p \Vdash_{\mu} \forall x \varphi \geq 1$ .

*Proof.* The proof is an easy modification of the proof of Corollary 10.

## 4 Classical Realisability and Random Realisability

In this section, we will study the relationship between classical and random realisability. In particular we will show that the two notions do not coincide.

We start by proving that classical realisability and  $\mu$ -realisability agree on pretty  $\Sigma_1$  sentences and that therefore  $\mu$ -realisability for pretty  $\Sigma_1$  sentences coincides with truth.

**Theorem 12.** *Let  $\varphi$  be a pretty  $\Sigma_1$  sentence in the language of arithmetic. Then, there are two computable functions  $P_\mu$  and  $P_\mu^{-1}$  such that for every  $p$ , (i)  $p \Vdash \varphi$  implies  $P_\mu(p, \varphi) \Vdash_\mu \varphi \geq 1$ , and (ii)  $p \Vdash_\mu \varphi$  implies  $P_\mu^{-1}(p, \varphi) \Vdash \varphi$ . Therefore a pretty  $\Sigma_1$  formula is true if and only if it is  $\mu$ -realised. The same result holds for universal  $\Pi_1$  sentences.*

*Proof.* We define  $P_\mu$  and  $P_\mu^{-1}$  by recursion on  $\varphi$  and will prove that they have the desired properties. We first define  $P_\mu$  and  $P_\mu^{-1}$  on  $\Delta_0$  formulas.

(1)  $\varphi$  is atomic: in this case realisability and  $\mu$ -realisability have the same realisers. So we can just let  $P_\mu$  and  $P_\mu^{-1}$  be the identity on atomic formulas.

(2) If  $\varphi \equiv \psi_0 \wedge \psi_1$  where  $\psi_0$  and  $\psi_1$  are  $\Delta_0$ .

First assume that  $p \Vdash_\mu \psi_0 \wedge \psi_1$ . Then,  $\mu(C_{p,\varphi}) > 0$  and for every  $u \in C_{p,\varphi}$  we have  $(p^u(0), u) \Vdash_{\mathcal{O}} \psi_0$  and  $(p^u(1), u) \Vdash_{\mathcal{O}} \psi_1$ . Let  $p(i)$  be the program that for every oracle  $u$  just returns  $p^u(i)$  for  $i \in \{0, 1\}$ . Then for  $i \in \{0, 1\}$  we have  $p(i) \Vdash_\mu \psi_i$ . Let  $P_\mu^{-1}(p, \varphi)$  be the program that given input  $i$  computes  $g(p(i), \psi_i)$ . By inductive hypothesis we have that  $P_\mu^{-1}(p(i), \psi_i) \Vdash \psi_i$  for every  $i \in \{0, 1\}$  and therefore  $P_\mu^{-1}(p, \varphi)$  realises  $\varphi$  as desired.

On the other hand let  $p \Vdash \psi_0 \wedge \psi_1$ . Then  $p(i) \Vdash \psi_i$  for every  $i \in \{0, 1\}$ . Let  $P_\mu(p, \varphi)$  be the program that ignores the oracle and for every  $i \in \{0, 1\}$  returns  $P_\mu(p(i), \psi_i)$ . By inductive hypothesis we have that  $P_\mu(p(i), \psi_i) \Vdash_\mu \psi_i \geq 1$ . Note that  $\mu(C_{P_\mu(p(0), \psi_0), \psi_0} \cap C_{P_\mu(p(1), \psi_1), \psi_1}) = 1$  and that for all  $u \in C_{P_\mu(p(0), \psi_0), \psi_0} \cap C_{P_\mu(p(1), \psi_1), \psi_1}$  we have that  $(P_\mu(p(0), \psi_0), u) \Vdash_{\mathcal{O}} \psi_0$  and  $(P_\mu(p(1), \psi_1), u) \Vdash_{\mathcal{O}} \psi_1$ . But then, since  $P_\mu(p, \varphi)^u(i) = P_\mu(p(i), \psi_i)$  for every oracle  $u$  and every  $i \in \{0, 1\}$ , we have that  $P_\mu(p, \varphi) \Vdash_\mu \psi_0 \wedge \psi_1 \geq 1$ .

(3) If  $\varphi \equiv \psi_0 \vee \psi_1$  where  $\psi_0$  and  $\psi_1$  are  $\Delta_0$ .

First assume that  $p \Vdash_\mu \psi_0 \vee \psi_1$ . Then,  $\mu(C_{p,\varphi}) > 0$  and for every  $u \in C_{p,\varphi}$  we have that  $(p^u(1), u) \Vdash_{\mathcal{O}} \psi_{p^u(0)}$ . Let  $p(1)$  be the program that for every oracle  $u$  returns  $p^u(1)$ . Then there is  $i \in \{0, 1\}$  such that  $p(1) \Vdash_\mu \psi_i$  by the proof of Corollary 10. Let  $P_\mu^{-1}(p, \varphi)$  be the program that for does the following: starts by running in parallel two instances of the program of Corollary 4, one with input  $\psi_0$  and one with input  $\psi_1$ . By inductive hypothesis note that at least one of the two instances will halt. Let  $i \in \{0, 1\}$  be such that the  $\psi_i$  instance halted first. Then, if the input is 0, the program halts with output  $P_\mu^{-1}(p(0), \psi_i)$ , and if the input is 1 the program returns  $i$ .

Now let  $p \Vdash \psi_0 \vee \psi_1$  where  $\psi_0$  and  $\psi_1$  are  $\Delta_0$ . Then  $p(1) \Vdash \psi_{p(0)}$ . Let  $P_\mu(p, \varphi)$  be the program that if the input is 0 halts with output  $P_\mu(p(1), \psi_{p(0)})$ , and if the input is 1 the program returns  $p(0)$ .

By inductive hypothesis we have that  $P_\mu(p(1), \psi_{p(0)}) \Vdash_\mu \psi_{p(0)} \geq 1$ . But then, by the proof of Corollary 11 since for every  $u$ ,  $P_\mu(p, \varphi)^u(1) = P_\mu(p(1), \psi_{p(0)})$ , and  $P_\mu(p, \varphi)(0) = p(0)$ , we have  $P_\mu(p, \varphi) \Vdash_\mu \psi_0 \vee \psi_1 \geq 1$  as desired.

(4) If  $\varphi \equiv \psi_0 \rightarrow \psi_1$  where  $\psi_0$  and  $\psi_1$  are  $\Delta_0$ . First assume that  $p \Vdash_\mu \psi_0 \rightarrow \psi_1$ . Let  $P_\mu^{-1}(p, \varphi)$  be the program that does the following: for every input returns



the code of the instance of the program in Corollary 4 with input  $\psi_1$ . Note that, if  $\psi_0$  is realisable, then by inductive hypothesis is  $\mu$ -realisable, by assumptions  $\psi_1$  is  $\mu$ -realisable and by inductive hypothesis  $\psi_1$  is realisable. In this case for every  $s$  we have that  $P_\mu^{-1}(p, \varphi)(s) \Vdash \psi_1$  and therefore  $P_\mu^{-1}(p, \varphi)$  is a realiser of  $\psi_1$ . On the other hand, if  $\psi_1$  is not realisable, then any natural number realises  $\varphi$ , so  $P_\mu^{-1}(p, \varphi)$  is again a realiser of  $\varphi$ .

Now let  $p \Vdash \psi_0 \rightarrow \psi_1$  where  $\psi_0$  and  $\psi_1$  are  $\Delta_0$ . Then for every realiser  $s$  of  $\psi_0$  we have that  $p(s) \Vdash \psi_1$ . Let  $P_\mu(p, \varphi)$  be the code of the program that for every input  $s$  and every oracle returns  $P_\mu(p(P_\mu^{-1}(s, \psi_0)), \psi_1)$ . By inductive hypothesis if  $s$  is a  $\mu$ -realiser of  $\psi_0$ , so  $P_\mu^{-1}(s, \psi_0)$  is a realiser of  $s$ . By assumption  $p(P_\mu^{-1}(s, \psi_0))$  is a realiser of  $\psi_1$ , and again by inductive hypothesis we have that  $P_\mu(p(P_\mu^{-1}(s, \psi_0), \psi_1)) \Vdash_\mu \psi_1 \geq 1$ . But then by Corollary 11 we have that  $P_\mu(p, \varphi) \Vdash_\mu \varphi \geq 1$  as desired.

(5) we omit the bounded quantifier cases because they are analogous to the conjunction and disjunction cases.

Now we extend the definition to pretty  $\Sigma_1$  formulas.

(6) If  $\varphi \equiv \exists x \psi$  where  $\psi$  is pretty  $\Sigma_1$ . First assume that  $p \Vdash_\mu \exists x \psi$ . Then, by Corollary 10 there must be  $n \in \mathbb{N}$  such that  $p(0) \Vdash_\mu \psi(n)$  therefore, by inductive hypothesis,  $\psi(n)$  is realised. Let  $P_\mu^{-1}(p, \varphi)$  be the program that does the following: run in parallel all the instances of the program of Corollary 4 with input  $\psi(n)$  with  $n \in \mathbb{N}$ . By inductive hypothesis note that one of these instances must halt. Let  $i \in \mathbb{N}$  be the least such that the  $\psi(i)$  instance halts. Then, if the input is 0, the program returns  $P_\mu^{-1}(p(0), \psi(i))$  and if it is 1, the program returns  $i$ .

Note that, by inductive hypothesis, the program halts and returns a realiser of  $\varphi$ , as desired.

Now assume that  $p \Vdash \exists x \psi$ . Then  $p(0) \Vdash \psi(p(1))$ . Let  $f(p, \varphi)$  be the program that returns  $P_\mu(p(0), \psi(p(1)))$  if the input is 0 and  $p(1)$  if the input is 1. By inductive hypothesis  $P_\mu(p(0), \psi(p(1))) \Vdash_\mu \psi(p(1)) \geq 1$ . But then by Corollary 11 since for all  $u$  we have  $P_\mu(p, \varphi)^u(0) = f(p(0), \psi(p(1)))$  and  $P_\mu(p, \varphi)^u(1) = p(1)$ , we have that  $P_\mu(p, \varphi) \Vdash_\mu \exists x \psi \geq 1$  as desired.

(7) If  $\varphi \equiv \forall x < n \psi$ , where  $\psi$  is pretty  $\Sigma_1$ . First assume that  $p \Vdash_\mu \forall x < n \psi$ . Then, by Corollary 10, for every natural number  $m < n$  and for every program  $s$  we have that  $p(m) \Vdash_\mu \psi(m)$  and by inductive hypothesis  $\psi(m)$  is realised. Let  $P_\mu^{-1}(p, \varphi)$  be the program that for every  $m$  returns a program that for every input if  $m < n$  returns  $P_\mu^{-1}(p(m), \psi(m))$  and returns 0 otherwise. For all  $m < n$ , by inductive hypothesis we have that  $P_\mu^{-1}(p(m), \psi(m)) \Vdash \psi(m)$  and therefore  $P_\mu^{-1}(p, \varphi)$  is a realiser of  $\varphi$  as desired.

Now assume that  $p \Vdash \forall x < n \psi$ . Then for all  $m < n$  and  $s$  we have that  $p(m)(s) \Vdash \psi(m)$ . Let  $P_\mu(p, \varphi)$  be the program that ignores the oracle and for every  $m$  returns a program that given  $q$  as input, if  $m < n$  then returns  $P_\mu(p(m)(q), \psi_1)$  otherwise returns 0. Now note that for every  $m < n$  and for

every program  $q$  we have that  $P_\mu(p(m)(q), \psi_1) \Vdash_\mu \psi(m) \geq 1$ . But then for every  $m < n$ , every  $q$ , and every  $u \in 2^\omega$  we have that  $(P_\mu(p, \varphi)^u(m))^u(q) \Vdash_\mu \psi \geq 1$ , and therefore by Corollary 11 we have  $P_\mu(p, \varphi) \Vdash_\mu \varphi \geq 1$  as desired.

Finally we extend the  $\Delta_0$  case to universal  $\Pi_1$  formulas.

(8) If  $\varphi \equiv \forall x\psi$  where  $\psi$  is universal  $\Pi_1$ . First assume that  $p \Vdash_\mu \forall x\psi$ . Let  $P_\mu^{-1}(p, \varphi)$  be the program that for all  $n$  runs  $P_\mu(P_\mu^{-1}(p(n), \psi(n)))$ . By Corollary 10 and the inductive hypothesis  $P_\mu^{-1}(p(n), \psi(n))$  is a realiser of  $\psi(n)$ . Therefore  $P_\mu^{-1}(p, \varphi)$  is a realiser of  $\forall x\psi$  as desired.

Now assume that  $p \not\Vdash_\mu \forall x\psi$ . Let  $P_\mu(p, \varphi)$  be the program that for every  $n$  and for every oracle returns  $P_\mu(p(n), \psi(n))$ . Once more by inductive hypothesis for all  $n$  and all  $\mu(C_{P_\mu(p(n), \psi(n)), \psi(n)}) = 1$  but then  $\mu(\bigcap_{n \in \mathbb{N}} C_{P_\mu(p(n), \psi(n)), \psi(n)}) = 1$  and  $P_\mu(p, \varphi) \Vdash_\mu \varphi \geq 1$  as desired.

The second part of the statement follows from Lemma 3.

**Corollary 13.** *Let  $\varphi$  be any false pretty  $\Sigma_1$  sentence in the language of arithmetic. Then  $(p, u) \Vdash_\circ \varphi \rightarrow \perp$  and  $p \Vdash_\mu \varphi \rightarrow \perp \geq 1$  for every  $p$  and  $u$ . The same holds for universal  $\Pi_1$  formulas.*

*Proof.* By Theorem 12 every  $\mu$ -realisable pretty  $\Sigma_1$  (universal  $\Pi_1$ ) sentence  $\varphi$  is true. Therefore  $\varphi$  cannot be  $\mu$ -realised and every program is going to  $\mu$ -realise  $\varphi \rightarrow \perp$ , which means that for all  $p$  and for all  $u$  we have  $p \Vdash_\mu \varphi \rightarrow \perp$  and  $(p, u) \Vdash_\circ \varphi \rightarrow \perp$  as desired.

We are now ready to prove the main result of this section, namely that  $\mu$ -realisability and classical realisability do not coincide. This result is surprising given that by Sacks's theorem [1, Corollary 8.12.2] functions that are computable with a non-null set of oracles are computable by a classical Turing machine.

**Theorem 14.** *There is a sentence  $\varphi$  in the language of arithmetic that is randomly realisable but not realisable.*

*Proof.* Let  $\varphi$  be the sentence “For all  $k$  there is  $n$  such that for all  $\ell$  the execution of  $p_k(k)$  does not stop in at most  $\ell$  steps or  $p_k(k) \neq n$ ” and let  $\psi(k)$  be the sentence “There is  $n$  such that for all  $\ell$  the execution of  $p_k(k)$  does not stop in at most  $\ell$  steps or  $p_k(k) \neq n$ ”.

A classical realiser for  $\varphi$  would be a program that computes a total function that, for every code  $k$  of a program, returns a natural number which is not the output of  $p_k(k)$ . By diagonalization, such a program cannot exist: If  $p_k$  was such a program, then it would follow that for every  $n \in \omega$  we have that  $p_k(k) = n \Leftrightarrow p_k(k) \neq n$ .

Now we want to show that  $\varphi$  is randomly realisable.

Fix any realiser  $s$ . Let  $p$  be the program that given an oracle  $u \in 2^\omega$ , a natural number  $k$ , and  $i \in \{0, 1\}$  does the following<sup>5</sup>: Let  $p^u(k)(i) = u \upharpoonright (k+1)$  if  $i = 1$

<sup>5</sup> Here, we do not distinguish between the finite sequence  $u \upharpoonright (k+1)$  and the natural number coding it.

and  $p^u(k)(i) = p'$  if  $i = 0$  where  $p'$  is the program that ignores the oracle and does the following:

On input  $\ell$ ,  $p'$  checks whether  $p_k(k)$  stops in  $\ell$  steps. If not, then  $p'(\ell)$  is the code of a program that returns 0 on input 0 and  $s$  on input 1. Otherwise  $p'(\ell)$  is the code of a program that returns 1 on input 0 and on input 1 looks for an  $\mu$ -realiser of the  $\Delta_0$  formula expressing the fact that “ $p_k(k) \neq u \upharpoonright (k+1)$ ” by running the algorithms in Lemma 3 and Theorem 12.

Now, for every  $k \in \omega$  and  $u \in 2^\omega$  we have two cases:

$p_k(k)$  does not halt: then we have that  $p^u(k)(1) = u \upharpoonright (k+1)$  and  $p^u(k)(0) = p'$ . Since  $p_k(k)$  does not halt, we have  $p'(\ell)(0) = 1$  and  $p'(\ell)(1) = s$  for every  $\ell$ . Moreover, by Corollary 13  $(s, u) \Vdash_{\mathcal{O}}$  “ $p_k(k)$  does not halt in  $\ell$  steps” and therefore  $(p^u(k), u) \Vdash_{\mathcal{O}} \psi(k)$ .

$p_k(k)$  halts: then we have that  $p^u(k)(1) = u \upharpoonright (k+1)$  and  $p^u(k)(0) = p'$ . Let  $\ell$  be such that  $p_k(k)$  halts in at most  $\ell$  steps. Then,  $p'(\ell)(0) = 0$ . Moreover, note that if the output of  $p_k(k)$  is not the same as the first  $k$  bits of the oracle then  $(p'(\ell)^u(1), u) \Vdash_{\mathcal{O}} u \upharpoonright (k+1) \neq p_k(k)$ .

We only need to show that  $\mu(C_{p,\varphi}) > 0$ . To see this, it is enough to note that the set of  $u$  such that  $p_k(k) \neq u \upharpoonright (k+1)$  has measure  $\geq 1 - \frac{1}{2^{(k+1)}}$ . Therefore,  $\mu(C_{p,\varphi}) = \prod_{k \in \mathbb{N}} (1 - \frac{1}{2^{(k+1)}}) > 0$  as desired.

**Corollary 15.** *There is a sentence in the language of arithmetic which is realisable but not randomly realisable.*

*Proof (Corollary 15).* It is enough to consider the sentence  $\varphi \rightarrow \perp$  where  $\varphi$  is the sentence in the proof of Theorem 14. The sentence is trivially realised since  $\varphi$  is not realised. Moreover, the sentence is not  $\mu$ -realised since  $\varphi$  is  $\mu$ -realised and  $\perp$  is not  $\mu$ -realised.

## 5 Soundness & Arithmetic

In this section, we study the logic and arithmetic of  $\mu$ -realisability. We first observe that, in a certain sense, the Law of Excluded Middle is not  $\mu$ -realisable.

**Lemma 16.** *There is  $\varphi$  such that  $\forall x(\varphi(x) \vee \neg\varphi(x))$  is not  $\mu$ -realisable.*

*Proof.* Let  $\varphi(x)$  be the formula expressing the fact that the program  $p_x(x)$  halts. Assume that  $\forall x(\varphi(x) \vee \neg\varphi(x))$  is randomly realised. Then, there is a program  $p$  such that  $p \Vdash_{\mu} \forall x(\varphi(x) \vee \neg\varphi(x))$ . Therefore,  $p$  computes the halting problem for a set of oracles of measure  $> 0$ . But this directly contradicts Sacks' theorem [1, Corollary 8.12.2].

We now show that  $\mu$ -realisability is preserved by the inference rules of first-order intuitionistic proof calculus.

First, we need to fix what it means for  $\varphi$  to be  $\mu$ -realizable when  $x$  occurs freely in  $\varphi$ : This is defined to mean the same as the  $\mu$ -realisability of  $\forall x\varphi$ .

**Definition 17 (Intuitionistic Calculus).** *Inference rules are:*

- MP : from  $\varphi$  and  $\varphi \rightarrow \psi$  infer  $\psi$   
 $\forall$  - GEN : from  $\psi \rightarrow \varphi$  infer  $\psi \rightarrow (\forall x \varphi)$ , if  $x$  is not free in  $\psi$ .  
 $\exists$  - GEN : from  $\varphi \rightarrow \psi$  infer  $(\exists x \varphi) \rightarrow \psi$ , if  $x$  is not free in  $\psi$ .

*The axioms are*

- THEN - 1 :  $\varphi \rightarrow (\chi \rightarrow \varphi)$   
 THEN - 2 :  $(\varphi \rightarrow (\chi \rightarrow \psi)) \rightarrow ((\varphi \rightarrow \chi) \rightarrow (\varphi \rightarrow \psi))$   
 AND - 1 :  $\varphi \wedge \chi \rightarrow \varphi$   
 AND - 2 :  $\varphi \wedge \chi \rightarrow \chi$   
 AND - 3 :  $\varphi \rightarrow (\chi \rightarrow (\varphi \wedge \chi))$   
 OR - 1 :  $\varphi \rightarrow \varphi \vee \chi$   
 OR - 2 :  $\chi \rightarrow \varphi \vee \chi$   
 OR - 3 :  $(\varphi \rightarrow \psi) \rightarrow ((\chi \rightarrow \psi) \rightarrow ((\varphi \vee \chi) \rightarrow \psi))$   
 FALSE :  $\perp \rightarrow \varphi$   
 PRED - 1 :  $(\forall x \varphi(x)) \rightarrow \varphi(t)$ , if the term  $t$  is free for substitution  
                   for the variable  $x$  in  $\varphi$   
 PRED - 2 :  $\varphi(t) \rightarrow (\exists x \varphi(x))$ , with the same restriction as for PRED - 1.

*Proof (Theorem 18).* We show that (i) all instantiations of the axioms of intuitionistic first-order calculus are  $\mu$ -realisable and (ii) the set of  $\mu$ -realizable statements is closed under modus ponens,  $\forall$ -GEN and  $\exists$ -GEN.

We start with (i).

THEN-1: A  $\mu$ -realiser for an instance of  $\varphi \rightarrow (\chi \rightarrow \varphi)$  needs to turn any given  $\mu$ -realiser  $r$  for  $\varphi$  into one for  $\chi \rightarrow \varphi$ . The  $\mu$ -realiser for  $\chi \rightarrow \varphi$  works by simply returning  $r$  for any input.

THEN-2: Here, we are given a  $\mu$ -realiser  $r$  for  $\varphi \rightarrow (\chi \rightarrow \psi)$  and our goal is to turn any  $\mu$ -realiser  $p$  for  $(\varphi \rightarrow \chi)$  into a  $\mu$ -realiser  $q$  for  $\varphi \rightarrow \psi$ . Given  $r$  and  $p$ ,  $q$  works as follows: Given a  $\mu$ -realiser  $s$  for  $\varphi$ , first use  $r$  to compute from  $s$  a  $\mu$ -realiser  $t$  for  $\chi \rightarrow \psi$  with positive probability; moreover, use  $p$  to compute from  $s$  a  $\mu$ -realiser for  $u \chi$  with positive probability. Then apply  $t$  to  $u$ .

AND-1 works by projecting the  $\mu$ -realiser for  $\varphi \wedge \chi$  to the first component, AND-2 by projecting to the second component.

AND-3: We need to turn any  $\mu$ -realiser  $p$  for  $\varphi$  into a  $\mu$ -realiser  $q$  for  $\chi \rightarrow (\varphi \wedge \chi)$  with positive probability. Let  $p$  be given. Also, let a  $\mu$ -realiser  $r$  for  $\chi$  be given. Now,  $q$  works as follows: For a given oracle  $x$ , let  $x = x_0 \oplus x_1$ , where, for real numbers  $a$  and  $b$ ,  $a \oplus b$  denotes the join of  $a$  and  $b$ , i.e.,  $2i \in a \oplus b$  iff  $i \in a$  and  $2i + 1 \in a \oplus b$  iff  $i \in b$ . Now  $(q^x(0), x)$  runs  $(p^{x_0}(0), x_0)$  while  $(q^x(1)(0), x)$  runs  $(r^{x_1}, x_1)$ .

OR-1 works by, given a  $\mu$ -realiser  $r$  for  $\varphi$ , sending 0 to 0 and 1 to  $r$ , OR-2 by sending 0 to 1 and 1 to  $r$ .

OR-3: We need to turn any  $\mu$ -realiser  $p$  for  $\varphi \rightarrow \psi$  into one for  $((\chi \rightarrow \psi) \rightarrow ((\varphi \vee \chi) \rightarrow \psi))$  with positive probability. Let  $q$  be a  $\mu$ -realiser for  $\chi \rightarrow \psi$ , and let  $r$  be a  $\mu$ -realiser for  $\varphi \vee \chi$ . Now, the sets  $S_0, S_1$  of oracles relative to which  $r$  realizes  $\varphi$  or  $\chi$ , respectively, are measurable, and as their union has positive measure, at least one of the sets  $S_0$  and  $S_1$  has positive measure. Thus, for a positive measure set  $S$  of oracles  $u$ ,  $r^u(0)$  will terminate with output  $i \in \{0, 1\}$  such that  $S_i$  has positive measure, so that  $(r^u(1), u)$  will be an  $O$ -realiser of  $\chi$  (if  $i = 0$ ) or  $\psi$  (if  $i = 1$ ), respectively. Let us denote by  $r(1)$  the program that, on oracle  $u$ , runs the program with index  $r^u(1)$  in the oracle  $u$ . With positive probability,  $r(1)$  will be an  $O$ -realiser of  $\varphi$  (if  $i = 0$ ) or  $\psi$  (if  $i = 1$ ). Now we proceed as follows: Given  $u$ , first compute  $r^u(0)$ . If this is 0, apply  $p$  to  $r(1)$ . If it is 1, apply  $q$  to  $r(1)$ . With positive probability, it then happens that  $p$  is applied to a  $\mu$ -realiser of  $\varphi$  or that  $q$  is applied to a  $\mu$ -realiser of  $\chi$ . In both cases, we obtain a  $\mu$ -realiser of  $\psi$ . Thus, we obtain a  $\mu$ -realiser of  $\psi$  with positive probability, as desired.

FALSE is  $\mu$ -realized by any program, as  $\perp$  does not have  $\mu$ -realisers.

PRED-1: Here,  $t$  will just be a natural number. Let  $r$  be a  $\mu$ -realiser for  $\forall x\varphi(x)$ . Let an oracle  $u$  be given, and suppose that  $r$  works for  $u$  (i.e.,  $(r, u) \Vdash_O \forall x\varphi(x)$ ), which happens for all  $u$  from a set of positive measure. For each such  $u$ ,  $(r^u(t), u)$  will be an  $O$ -realiser for  $\varphi(t)$  by definition. Thus, the program  $r(t)$  that, for given  $u$ , runs the program with index  $r^u(t)$  in the oracle  $u$  is a  $\mu$ -realiser for  $\varphi(t)$ .

PRED-2: Let  $r$  be a  $\mu$ -realiser for  $\varphi(t)$ . Then a  $\mu$ -realiser  $p$  for  $\exists x\varphi(x)$  works by letting  $p^u(1)$  output  $t$  and letting  $p^u(0)$  output  $r$  for every  $u$ .

Now for (ii).

(1) (MP) If  $\varphi$  and  $\varphi \rightarrow \psi$  are  $\mu$ -realizable, then so is  $\psi$ .

Suppose that  $p$   $\mu$ -realizes  $\varphi$  and that  $q$   $\mu$ -realizes  $\varphi \rightarrow \psi$ . Pick a real number  $x$  such that  $(q, x)$  realizes  $\varphi \rightarrow \psi$  and run  $q^x(p)$ . By definition, the output is a  $\mu$ -realiser for  $\psi$ .

(2)  $\forall$ -GEN

Let  $p$  be a  $\mu$ -realiser for  $\psi \rightarrow \varphi$ , and let  $q$   $\mu$ -realize  $\psi \rightarrow (\forall x\varphi)$ , where  $x$  does not occur freely in  $\psi$  but (possibly) in  $\varphi$ . If  $x$  does not occur freely in  $\psi$ , the claim is trivial since then  $\forall x\varphi$  is  $\mu$ -realizable if and only if  $\varphi$  is. We are given  $n \in \omega$ , our goal is to compute a realiser for  $\varphi(n)$ . Pick some oracle  $y$  such that  $(p, y)$  realizes  $\psi \rightarrow \varphi$ . Note that this means that  $(p, y)$  computes a realiser for  $\psi \rightarrow \varphi(n)$  from any given  $n \in \omega$ . Now run this realiser in the input  $n$ ; by definition, the output will be a  $\mu$ -realiser for  $\varphi(n)$ , as desired.

(3)  $\exists$ -GEN

Let  $p$  be a  $\mu$ -realiser for  $\varphi \rightarrow \psi$  and let  $q$  be a  $\mu$ -realiser for  $(\exists x\varphi) \rightarrow \psi$ , where  $x$  is not free in  $\psi$ . Pick oracles  $y$  and  $z$  such that  $(q, y)$  realizes  $(\exists x\varphi) \rightarrow \psi$  and  $(p, z)$  realizes  $\varphi \rightarrow \psi$ . Thus,  $(q^y(0), y)$  realizes  $\varphi(n)$ , where  $n$  is the output of  $q^y(1)$ . Recall that  $p$  is a  $\mu$ -realiser for  $\forall x(\varphi \rightarrow \psi)$ . Thus,  $p^z(n)$  is the index of a program  $r$  that turns  $\mu$ -realisers for  $\varphi(n)$  into  $\mu$ -realisers for  $\psi$ . Consequently, running  $p^z(n)$  on the input  $q(1)$  yields a  $\mu$ -realiser for  $\psi$ .

**Theorem 18 (Soundness).** *The set of  $\mu$ -realizable statements is closed under the rules of intuitionistic first-order calculus.*

It is a classical result that the axioms of Heyting Arithmetic are realisable, see [6, Theorem 1]. We show that only a fragment of HA is  $\mu$ -realisability. Let  $\text{HA}^-$  denote the axioms of Peano arithmetic without the induction schema. As usual, *Heyting arithmetic* HA is the theory obtained from adding the induction schema to  $\text{HA}^-$ . We say that a set of formulas  $\Gamma$  is  $\mu$ -realised if  $\varphi$  is  $\mu$ -realised for all  $\varphi \in \Gamma$ .

Since all the axioms except for the induction schema are universal  $\Pi_1$  statements, it follows by Theorem 12 that the axioms of  $\text{HA}^-$  are all  $\mu$ -realised.

**Theorem 19.** *The set  $\text{HA}^-$  is  $\mu$ -realised.*

Contrary to the classical case the induction schema fails for  $\mu$ -realisability.

**Theorem 20.** *The induction schema is not  $\mu$ -realised.*

*Proof.* Let  $\varphi(x)$  be the formula expressing the fact that “Every program with code  $i < x$  halts or does not halt”. By the proof of Lemma 16,  $\varphi$  is not  $\mu$ -realisable.

On the other hand, a  $\mu$ -realiser  $p(n)$  for  $\varphi(n)$  is given by a program that does the following: for every  $i < n$ ,  $p$  returns a program that if the  $i$ th element of the oracle is 1 returns 1 on input 0 and any number on input 1. While if the  $i$ th element of the oracle is 0 the program returns 0 on input 0 and on input 1 starts building a realiser of “the program  $i$  halts” using the algorithm in Lemma 4; if it finds one, it runs the algorithm in Theorem 12 to compute the desired  $\mu$ -realiser.

It is not hard to see that the algorithm works with probability  $\frac{1}{2^n}$ . Thus, to realize the implication  $\varphi(n) \rightarrow \varphi(n+1)$ , we can ignore the  $\mu$ -realiser for  $\varphi(n)$  and just output  $p(n)$ . So the premise of the instance of the induction schema is  $\mu$ -realised, while the conclusion is not.

Note that the proof of Theorem 20 heavily relies on the fact that the definition of  $\mu$ -realisability does not require any relationship between the measures of the set of oracles realising the antecedent of an implication and the set of oracles realising the consequent. We think that a modification of this definition could lead to a notion of probabilistic realisability that realises the induction schema.

Even though the axiom schema of induction is not  $\mu$ -realisable, one can prove that all  $\Delta_0$ -instances of the schema are realisable. Indeed, by Theorem 12 and the fact that if  $\varphi$  is a  $\Delta_0$  formula then  $\forall x\varphi(x, \bar{y})$  is a universal  $\Pi_1$  formula, we have the following:

**Corollary 21.** *The set  $\text{HA}^-$  together with the induction schema restricted to  $\Delta_0$  formulas is  $\mu$ -realisable.*

## 6 Big Realisability

In this section, we will consider other natural definitions of realisability arising from notions of *big sets of oracles* on the real numbers. More specifically, we will consider “almost sure realisability,” “comeagre realisability,” “interval-free realisability,” and “positive measure realisability.” It will turn out, however, that the first three are equivalent to standard realisability, while the final one coincides with truth. We begin with the following general definition.

**Definition 22.** *Let  $\mathcal{F}$  be a family of subsets of Cantor space  $2^\omega$ . We then define  $\mathcal{F}$ -realisability recursively as follows:*

1.  $p \Vdash_{\mathcal{F}} \perp$  never,
2.  $p \Vdash_{\mathcal{F}} n = m$  if and only if  $n = m$ ,
3.  $p \Vdash_{\mathcal{F}} \psi_0 \wedge \psi_1$  if and only if  $p(i) \Vdash_{\mathcal{F}} \psi_i$  for  $i < 2$ ,
4.  $p \Vdash_{\mathcal{F}} \psi_0 \vee \psi_1$  if and only if there is some  $O \in \mathcal{F}$  and some  $i < 2$  such that for every  $u \in O$ , we have  $p^u(0) = i$  and  $p^u(1) \Vdash_{\mathcal{F}} \psi_i$ ,
5.  $p \Vdash_{\mathcal{F}} \varphi \rightarrow \psi$  if and only if there is a set  $O \in \mathcal{F}$ , such that for every  $u \in O$  and  $s \Vdash_{\mathcal{F}} \varphi$ , we have  $p^u(s) \Vdash_{\mathcal{F}} \psi$ ,
6.  $p \Vdash_{\mathcal{F}} \exists x \varphi$  if and only if there is some  $O \in \mathcal{F}$ , such that there is some  $n$  for all  $u \in O$  such that  $p^u(0) = n$  and  $p^u(1) \Vdash_{\mathcal{F}} \varphi(n)$ ,
7.  $p \Vdash_{\mathcal{F}} \forall x \varphi$  if and only if there is a set  $O \in \mathcal{F}$ , such that for every  $u \in O$  and  $n \in \mathbb{N}$  we have  $p^u(n) \Vdash_{\mathcal{F}} \varphi(n)$ .

From this definition, we derive the following notions of realisability: Let  $\mathcal{F}_{\text{if}}$  be the family of *co-interval-free* subsets of the Cantor space, i.e.  $X \in \mathcal{F}_{\text{cif}}$  if and only if  $X \in 2^\omega$  and there is no open interval  $I$  such that  $I \subseteq 2^\omega \setminus X$ , and  $\Vdash_{\text{cif}}$  denotes  $\mathcal{F}_{\text{cif}}$ -realisability. Let  $\mathcal{C}$  be the family of comeagre subsets of the Cantor space, then let  $\Vdash_{\mathcal{C}}$  denote  $\mathcal{C}$ -realisability. Let  $\mathcal{F}_{=1}$  be the family of subsets of the Cantor space that are of measure 1, and let  $\Vdash_{=1}$  denote  $\mathcal{F}_{=1}$ -realisability. Let  $\mathcal{F}_{>0}$  be the family of subsets of the Cantor space of positive measure, and  $\Vdash_{>0}$  denotes  $\mathcal{F}_{>0}$ -realisability. As before, we will write  $\Vdash_{\mathcal{F}} \varphi$  if and only if there is some realiser  $p$  such that  $p \Vdash_{\mathcal{F}} \varphi$ .

In what follows we will make use of the *bounded exhaustive search with  $p(n)$* , i.e. the following procedure. Given a program  $p$  (and possibly some input  $n$ ), do the following successively for all  $k \in \omega$ . Enumerate all 0-1-strings of length  $k$ . For each of these strings  $s$ , do the following: Run  $p^s(n)$  for  $k$  many steps. If the computation does not halt within that time (which implies in particular that at most the first  $k$  many bits of the oracle were requested), continue with the next  $s$  (if there is one, otherwise with  $(k + 1)$ ). If the computation halts with output  $x$  within that time, then the search terminates with output  $x$ .

The crucial property of this procedure, which is also contained in the proof idea of Sacks’ theorem [1, Corollary 8.12.2], is the following:

**Lemma 23.** *Let  $G \subseteq \omega$ ,  $n \in \omega$  and let  $p$  be a program. Suppose that there is a set  $S \subseteq 2^\omega$  such that  $2^\omega \setminus S$  is interval-free and  $p^u(n)$  terminates for all  $u \in S$  with output  $k \in G$ . Then the bounded exhaustive search with  $p(n)$  will terminate with output  $k \in G$ .*

*Proof.* Note that for every  $n$  and  $u \in S$  we have that  $p^u(n)$  terminates with output in  $G$ . So there is a finite initial segment  $s$  of  $u$  such that  $p^s(n)$  terminates with output  $p^u(n)$ . So, the bounded exhaustive search will halt.

Now, note that if the search halts on the string  $s$  with output  $k \in \omega$ , but  $k \notin G$ , then  $p^x(n) \downarrow k$  for all  $u \in N_s$ . But then,  $N_s \subseteq 2^\omega \setminus S$  which contradicts the fact that  $2^\omega \setminus S$  is interval free.

**Lemma 24.** *Let  $X \subseteq 2^\omega$  be a subset of Cantor space. If  $\mu(X) = 0$  or  $X$  is meagre, then  $X$  is interval-free.*

*Proof.* The first statement follows trivially from the fact that every non-empty open interval has positive measure. For the second statement, recall that meagre sets have empty interior by the Baire Category Theorem (cf. [2, Theorem 0.11]) and therefore contain no intervals.

**Theorem 25.** *Let  $\mathcal{F}$  be a family of subsets of Cantor space such that every  $X \in \mathcal{F}$  is co-interval-free. There are programs  $P_{\mathcal{F}}$  and  $P_{\mathcal{F}}^{-1}$  such that the following holds for all statements  $\varphi$ : (i) if  $p \Vdash \varphi$ , then  $P_{\mathcal{F}}(p, \varphi) \Vdash_{\mathcal{F}} \varphi$ , (ii) if  $p \Vdash_{\mathcal{F}} \varphi$ , then  $P_{\mathcal{F}}^{-1}(p, \varphi) \Vdash \varphi$ . Consequently,  $\varphi$  is realisable if and only if it is  $\mathcal{F}$ -realisable, and  $\Vdash, \Vdash_{\text{cif}}, \Vdash_{\mathcal{C}}$ , and  $\Vdash_{=1}$  coincide.*

*Proof.* We show both statements by simultaneous induction on the complexity of  $\varphi$  and simultaneously define  $P_{\mathcal{F}}$  and  $P_{\mathcal{F}}^{-1}$  by recursion on  $\varphi$ .

(1)  $\varphi$  is  $t_0 = t_1$  or  $t_0 \neq t_1$ . In this case,  $\mathcal{F}$ -realisers and realisers are the same, so the statement is trivial:  $P_{\mathcal{F}}$  and  $P_{\mathcal{F}}^{-1}$  just return the first component.

(2)  $\varphi$  is  $\psi_0 \wedge \psi_1$ .

Let  $r = (r_0, r_1)$  be a realiser for  $\varphi$  such that  $r_i$  realises  $\psi_i$  for  $i < 2$ . By induction hypothesis,  $P_{\mathcal{F}}(r_i, \psi_i)$  will return an  $\mathcal{F}$ -realiser for  $\psi_i$ . Hence,  $P_{\mathcal{F}}(r, \varphi)$  is the program that outputs  $P_{\mathcal{F}}(r_i, \psi_i)$  on input  $i$ . We obtain  $P_{\mathcal{F}}^{-1}$  in exactly the same way.

(3)  $\varphi$  is  $\psi_0 \vee \psi_1$ .

Let  $r$  be a realiser for  $\varphi$ , i.e.  $r(0)$  returns some  $i < 2$  and  $r(1) \Vdash \psi_i$ . By induction hypothesis, we have that  $P_{\mathcal{F}}(r(1), \psi_i) \Vdash_{\mathcal{F}} \psi_i$ . Hence,  $P_{\mathcal{F}}(r, \varphi)$  is the program that returns  $i$  on input 0 and  $P_{\mathcal{F}}(r(1), \psi_i)$  on input 1.

Conversely, let  $r$  be an  $\mathcal{F}$ -realiser for  $\varphi$ . Then there are some  $i < 2$  and  $O \in \mathcal{F}$  such that for all  $u \in O$ ,  $r^u(0) = i$  and  $r^u(1) \Vdash_{\mathcal{F}} \psi_i$ . Hence, let  $P_{\mathcal{F}}^{-1}(r, \varphi)$  be the program that executes a bounded exhaustive search with  $r^u(0)$ , which terminates by Lemma 23 in some  $i < 2$ , and then returns  $i$  on input 0, and  $P_{\mathcal{F}}^{-1}(r, \psi_i)$  on input 1. Then  $P_{\mathcal{F}}^{-1}(r) \Vdash \varphi$ .

(4)  $\varphi$  is  $\psi_0 \rightarrow \psi_1$ .

Let  $r \Vdash \varphi$ . Then  $r$  is a program that, given a realiser  $r_0 \Vdash \psi_0$ , returns a realiser  $r_1 \Vdash \psi_1$ . Let  $r'_0 \Vdash_{\mathcal{F}} \psi_0$ . By induction hypothesis,  $P_{\mathcal{F}}^{-1}(r'_0, \psi_0) \Vdash \psi_0$ . Hence,  $r(P_{\mathcal{F}}^{-1}(r'_0, \psi_0)) \Vdash \psi_1$  and  $P_{\mathcal{F}}(r(P_{\mathcal{F}}^{-1}(r'_0, \psi_0)), \psi_1) \Vdash_{\mathcal{F}} \psi_1$ . Therefore, let



$P_{\mathcal{F}}(r, \varphi)$  be the program that takes a realiser  $r'_0 \Vdash \psi_0$  as input and returns  $P_{\mathcal{F}}(r(P_{\mathcal{F}}^{-1}(r'_0, \psi_0)), \psi_1)$ .

The proof for the other direction is symmetric by exchanging the roles of  $P_{\mathcal{F}}$  and  $P_{\mathcal{F}}^{-1}$ .

(5)  $\varphi$  is  $\exists x\psi(x)$ .

Let  $r \Vdash \exists x\psi(x)$ . Then  $r(0) = n$  and  $r(1) \Vdash \psi(n)$ . By induction hypothesis, it follows that  $P_{\mathcal{F}}(r(1), \psi) \Vdash_{\mathcal{F}} \psi(n)$ . So let  $P_{\mathcal{F}}(r, \varphi)$  be the program that output  $n$  on input 0, and  $P_{\mathcal{F}}(r(1), \psi)$  on input 1. Then,  $P_{\mathcal{F}}(r, \varphi) \Vdash_{\mathcal{F}} \varphi$ .

Conversely, let  $r \Vdash_{\mathcal{F}} \exists x\psi(x)$ . Then there is some  $O \in \mathcal{F}$  and  $n \in \omega$  such that  $r^u(0) = n$  and  $r^u(1) \Vdash_{\mathcal{F}} \psi(n)$ . By induction hypothesis,  $P_{\mathcal{F}}^{-1}(r^u(1), \psi) \Vdash \psi(n)$ . Define  $P_{\mathcal{F}}^{-1}(r, \varphi)$  to be the following program: First, start a bounded exhaustive search with  $r(0)$ . By Lemma 23 this search must terminate with output  $n$ . Return  $n$  on input 0, and return  $P_{\mathcal{F}}^{-1}(r^u(1), \psi)$  on input 1. Then  $P_{\mathcal{F}}^{-1}(r, \varphi) \Vdash \exists x\psi(x)$ .

(6)  $\varphi$  is  $\forall x\psi(x)$ .

Let  $r \Vdash \forall x\psi(x)$ . Then  $r(n) \Vdash \psi(n)$  for every  $n \in \omega$ . Let  $P_{\mathcal{F}}(r, \varphi)$  be the program that, given  $n \in \omega$ , returns  $P_{\mathcal{F}}(r(n), \psi)$ . With the induction hypothesis, it follows that  $P_{\mathcal{F}}(r, \varphi) \Vdash_{\mathcal{F}} \varphi$ .

Conversely, let  $r \Vdash_{\mathcal{F}} \forall x\psi(x)$ . Then there is some  $O \in \mathcal{F}$  such that for every  $u \in O$  and  $n \in \mathbb{N}$  we have that  $r^u(n) \Vdash \psi(n)$ . Define  $P_{\mathcal{F}}^{-1}(r, \varphi)$  to be the following program: Start a bounded exhaustive search with  $r(n)$ . By Lemma 23, this search will terminate with  $r' \Vdash_{\mathcal{F}} \psi(n)$ . Then return  $P_{\mathcal{F}}^{-1}(r', \psi)$ , which, by induction hypothesis, is a realiser of  $\psi(n)$ . Hence,  $P_{\mathcal{F}}^{-1}(r, \varphi) \Vdash \psi$ .

**Theorem 26.** *Let  $\varphi$  be a formula. Then  $\Vdash_{>0} \varphi$  if and only if  $\varphi$  is true.*

*Proof.* The proof is an induction on the complexity of  $\varphi$ .

(1) If  $\varphi$  is atomic the statement follows by the definitions.

(2) Assume that  $\varphi \equiv \psi_0 \wedge \psi_1$ .

If  $\varphi$  is true then by inductive hypothesis there are  $p$  and  $q$  such that  $p \mathcal{F}_{>0}$ -realises  $\psi_0$  and  $q \mathcal{F}_{>0}$ -realises  $\psi_1$ . Let  $s$  be a sequence which starts with a code of  $p$  followed by a marker and by a code for  $q$  followed by a second marker. Then let  $t$  be the program that on input 0 returns the content of the oracle up to the first marker and on input 1 returns the content of the oracle between the first and the second marker. Note that for all  $u \in \mathbb{N}_s$ ,  $r^u(0) \Vdash_{>0} \psi_0$  and  $r^u(1) \Vdash_{>0} \psi_1$ . So,  $r \Vdash_{>0} \varphi$  as desired.

On the other hand if  $\varphi$  is  $\mathcal{F}_{>0}$ -realised then by definition both  $\psi_0$  and  $\psi_1$  are  $\mathcal{F}_{>0}$ -realised and the statement follows by the inductive hypothesis.

(3) Assume that  $\varphi \equiv \psi_0 \vee \psi_1$ .

If  $\varphi$  is true then by inductive hypothesis there is  $p$  such that  $p^u(1) \Vdash_{>0} \psi_{p^u(0)}$  for every  $u$  in some positive measure set  $O$ . Let  $s$  be a sequence which starts

with  $p^u(0)$  followed by a code for  $p^u(1)$  followed by a marker. Then let  $q$  be the program that on input 0 returns the content of the first bit of the oracle and on input 1 returns the content of the oracle from the second bit to the marker. Note that for all  $u \in \mathbb{N}_s$ ,  $q^u(1) \Vdash_{>0} \psi_{q^u(0)}$ . So,  $q \Vdash_{>0} \varphi$  as desired.

On the other hand if  $\varphi$  is  $\mathcal{F}_{>0}$ -realised then by definition at least one between  $\psi_0$  and  $\psi_1$  is  $\mathcal{F}_{>0}$ -realised and the statement follows by the inductive hypothesis.

(4) Assume that  $\varphi \equiv \psi_0 \rightarrow \psi_1$ .

Assume that  $\varphi$  is true. Then either  $\psi_1$  is true or  $\psi_0$  is false. If  $\psi_0$  is false then by inductive hypothesis is not  $\mathcal{F}_{>0}$ -realised and therefore any natural number will  $\mathcal{F}_{>0}$ -realise  $\varphi$ . If  $\psi_1$  is true, then by inductive hypothesis is  $\mathcal{F}_{>0}$ -realised by some program  $p$ . Let  $s$  be the sequence that starts with a code of  $p$  followed by a marker. Let  $q$  the program that for every  $n$  and every oracle returns the content of the oracle up to the first occurrence of the marker. Then for all  $u \in \mathbb{N}_s$  and for every  $n$  we have that  $q^u(n) \Vdash_{>0} \psi_1$ . So,  $q \Vdash_{>0} \varphi$  as desired.

On the other hand if  $\varphi$  is  $\mathcal{F}_{>0}$ -realised by some program  $p$ . If  $\psi_0$  is true then it is  $\mathcal{F}_{>0}$ -realised by some program  $q$ . Then there is a non-null set  $O$  such that for all  $u \in O$  we have that  $p^u(q) \Vdash_{>0} \psi_1$ . But then by inductive hypothesis  $\psi_1$  must be true.

(5) Assume that  $\varphi \equiv \exists x\psi$ .

Assume that  $\varphi$  is true. Then for some  $n \in \mathbb{N}$  we have that  $\psi(n)$  is true. By inductive hypothesis there is  $p$  which  $\mathcal{F}_{>0}$ -realises  $\psi(n)$ . Let  $s$  be a sequence starting with a code for  $n$  followed by a marker and then by the code of  $p$  followed by a marker. Let  $q$  be the program that on input 0 returns the content of the oracle up to the first marker, and on input 1 returns the content of the oracle between the first and second marker. Then for all  $u \in \mathbb{N}_s$  and for every  $n$  we have that  $q^u(1) \Vdash_{>0} \psi(q^u(0))$ . So,  $q \Vdash_{>0} \varphi$  as desired.

On the other hand if  $\varphi$  is  $\mathcal{F}_{>0}$ -realised by some program  $p$ . Then there is a non-null set  $O$  such that for all  $u \in O$  we have that  $p^u(1) \Vdash_{>0} \psi(p^u(0))$ . But then by inductive hypothesis  $\psi_1$  must be true.

(6) Assume that  $\varphi \equiv \forall x\psi$ .

Assume that  $\varphi$  is true. Then for all  $n \in \mathbb{N}$  we have that  $\psi(n)$  is true. Without loss of generality we can assume that the main operator of  $\psi$  is not a universal quantifier, the proof can be easily modified otherwise. Let  $q$  be the program that ignores the oracle and depending on the main connective of  $\psi$  does the following:

- if  $\psi$  is atomic  $q$  is just the constant function 1;
- if  $\psi$  is  $\psi_0 \wedge \psi_1$  then  $q(n)$  is the program  $r$  from the proof of case (2);
- if  $\psi$  is  $\psi_0 \vee \psi_1$  then  $q(n)$  is the program  $q$  from the proof of case (3);
- if  $\psi$  is  $\psi_0 \rightarrow \psi_1$  then  $q(n)$  is the program  $q$  from the proof of case (4);
- if  $\psi$  is  $\exists x\psi_0$  then  $q(n)$  is the program  $q$  from the proof of case (5);

By inductive hypothesis and by (2), (3), (4), and (5) of this proof we have that for all  $u \in 2^\omega$  and for every  $n$  we have that  $q^u(n) \Vdash_{>0} \psi(n)$ . So,  $q \Vdash_{>0} \varphi$  as desired.

On the other hand if  $\varphi$  is  $\mathcal{F}_{>0}$ -realised by some program  $p$ . Then there is a non-null set  $O$  such that for all  $u \in O$  we have that  $p^u(n) \Vdash_{>0} \psi(n)$ . But then by inductive hypothesis  $\varphi$  must be true.

## References

1. Downey, R., Hirschfeldt, D.: Algorithmic Randomness and Complexity. Theory and Applications of Computability, Springer New York (2010)
2. Kanamori, A.: The Higher Infinite: Large Cardinals in Set Theory from Their Beginnings. Springer Monographs in Mathematics, Springer (2008)
3. Kechris, A.: Classical Descriptive Set Theory, Graduate Texts in Mathematics, vol. 156. Springer (2012)
4. Kleene, S.C.: On the interpretation of intuitionistic number theory. J. Symbolic Logic **10**, 109–124 (1945)
5. Moschovakis, Y.: Descriptive Set Theory, Studies in Logic and the Foundations of Mathematics, vol. 100. Elsevier (1987)
6. Nelson, D.: Recursive functions and intuitionistic number theory. Transactions of the American Mathematical Society **61**(2), 307–368 (1947)
7. van Oosten, J.: Realizability: A historical essay. Math. Struct. Comput. Sci. **12**(3), 239–263 (2002)
8. Troelstra, A.S. (ed.): Metamathematical investigation of intuitionistic arithmetic and analysis. Lecture Notes in Mathematics, Vol. 344, Springer-Verlag, Berlin-New York (1973)
9. Turing, A.M.: 'intelligent machinery', national physical laboratory report. In: Meltzer, B., Michie, D. (eds.) Machine Intelligence 5. Edinburgh University Press (1969). (1948)