LAWVERE-TIERNEY TOPOLOGIES FOR COMPUTABILITY THEORISTS

TAKAYUKI KIHARA

ABSTRACT. In this article, we study the lattice of Lawvere-Tierney topologies on Hyland's effective topos. For this purpose, we introduce a new computability-theoretic reducibility notion, which is a common extension of the notions of Turing reducibility and generalized Weihrauch reducibility. Based on the work by Lee and van Oosten, we utilize this reducibility notion for providing a concrete description of the lattice of the Lawvere-Tierney topologies on the effective topos. As an application, we solve several open problems proposed by Lee and van Oosten. For instance, we show that there exists no minimal Lawvere-Tierney topology which is strictly above the identity topology on the effective topos.

1. INTRODUCTION

1.1. **Summary.** Our goal in this article is to accomplish a detailed analysis of the entire structure of "intermediate worlds" between "the world of computable mathematics" and "the world of set-theoretic mathematics." In [22], Hyland discovered the *effective topos* **Eff**, and proposed it as the *world of computable mathematics*. In topos theory, there is a notion called a *Lawvere-Tierney topology* (also known as a local operator or a geometric modality), and any topology j on a topos $\mathcal{E}$ yields a new subtopos $\mathcal{E}_j \hookrightarrow \mathcal{E}$ (see e.g. [28]). Indeed, Lawvere-Tierney topologies on $\mathcal{E}$ are in one-to-one correspondence with subtoposes of $\mathcal{E}$. The least topology is the identity topology Id that does not cause any change to the base topos. The largest topology is the indiscrete topology that contracts all truth-values to a single value, and the resulting degenerated topos may be thought of as the *world of inconsistent mathematics*. The next largest topology is the double negation $\neg\neg$. In the effective topos, the new topos **Eff**$_{\neg\neg}$ created from $\neg\neg$ is exactly the *world of set-theoretic mathematics*; that is, **Eff**$_{\neg\neg} \simeq$ **Set**. What this suggests is that analyzing the intermediate topologies between Id and $\neg\neg$ on the effective topos may correspond to exploring the intermediate worlds between computable mathematics and set-theoretic mathematics.

Under this perspective, a topology on the effective topos is a kind of data that indicate how much non-computability to add to the world. In other words, a topology plays the same role as an *oracle*. Indeed, Hyland [22] noticed that each Turing degree **d** has a corresponding topology $j_{\mathbf{d}}$ on the effective topos, which yields the *world of **d**-relatively computable mathematics*. However, this does not mean that we have exhausted all the topologies, and of course, there may be other topologies besides them. For instance, instead of a subset of $\mathbb{N}$ or a total function on $\mathbb{N}$, one can use a partial function as an

oracle. Not only that, but even a partial multi-valued function can be used as an oracle, and has a corresponding topology on the effective topos as we observe in this article. As another example, Pitts [37] found an intermediate topology that is not bounded by any Turing degree topology. This topology has properties that are far from any of the other topologies mentioned above. Remarkably, Lee-van Oosten [26] gave a concrete presentation of all topologies on the effective topos.

The first step of our work in this article is to capture the presentation of Lee-van Oosten [26] within the framework of *generalized Weihrauch reducibility* [21]. However, generalized Weihrauch reducibility (which involves a perfect information game) itself is insufficient to deal with all topologies, so we introduce an imperfect information game that incorporates some sort of nonuniform computation with advices. Coincidentally, it turns out that our notion is heavily related to another notion called *extended Weihrauch reducibility*, which is introduced in Bauer [2]. By viewing topologies in this way, it is possible, for example, to position the study of the structure of Lawvere-Tierney topologies on (equivalently, that of subtoposes of) a relative realizability topos as an extension of the Weihrauch-style analogue [6] of reverse mathematics (note, however, that this is by no means an extension of the standard reverse mathematics [42, 12, 13] at all; reverse mathematics has more to do with the internal logic, and more finitary aspects). By bringing the arguments on topologies/subtoposes into pure computability theory in this way, we solve some problems proposed in [25, 26].

While the notion of Lawvere-Tierney topology is originally studied in an abstract context, we develop our theory in the most intuitive and elementary way possible. We believe that it is important for the development of a theory to present it in a way that reduces prior knowledge of the theory as much as possible. For this purpose, we maintain appropriate notations and keep the discussion moving forward with concrete ideas. In Section 2, we introduce certain kinds of computable reduction games with imperfect information. By using these games, we also define a notion of computability-theoretic reduction for certain extended functions. In Section 3, we see that this reducibility notion characterizes the notion of Lawvere-Tierney topology on the effective topos, based on the idea in Lee-van Oosten [26]. In Section 4, by using the characterization, we solve some problems on topologies proposed in [25, 26]. For instance, we show that there exists no minimal Lawvere-Tierney topology which is strictly above the identity topology on the effective topos. Thus, in a certain sense, there is no world of non-computable mathematics which is closest to computable mathematics. In Section 5, we discuss a few other topologies, which has not been studied in the past. One corresponds to the world of computable mathematics with error probability ε, and the other to computable mathematics with error density ε (in the sense of lower asymptotic density).

1.2. **Notations.** In this article, we assume that the reader is familiar with elementary facts about computability theory. For the basics of computability theory, we refer the reader to [11, 33, 39, 43]. We use the following notations on strings: Let $\mathbb{N}^{<\mathbb{N}}$ be the set of all finite strings. For finite strings $\sigma, \tau \in \mathbb{N}^{<\mathbb{N}}$, we write $\sigma \preceq \tau$ if σ is an initial segment of τ, and write $\sigma \prec \tau$ if σ is a proper initial segment of τ. We also use the same notation even if τ is an infinite string, i.e., $\tau \in \mathbb{N}^{\mathbb{N}}$. For $\sigma \in \mathbb{N}^{<\mathbb{N}} \cup \mathbb{N}^{\mathbb{N}}$ and $\ell \in \mathbb{N}$, define $\sigma \upharpoonright \ell$ as the initial segment of σ of length ℓ. For finite strings $\sigma, \tau \in \mathbb{N}^{<\mathbb{N}}$, let $\sigma^\frown \tau$

be the concatenation of σ and τ. If τ is a string of length 1, i.e., τ is of the form $\langle n \rangle$ for some $n \in \mathbb{N}$, then $\sigma^\frown \langle n \rangle$ is abbreviated to $\sigma^\frown n$. Similarly, $\langle n \rangle^\frown \tau$ is abbreviated as $n^\frown \tau$.

A tree is a set $T \subseteq \mathbb{N}^{<\mathbb{N}}$ which is downward closed under $\preceq$. An element of a tree T is often called a node. The $\preceq$-least node (i.e., the empty string) of T is called the root, and a $\preceq$-maximal node of T is called a leaf. We always assume that $\mathbb{N}^{\mathbb{N}}$ is equipped with the standard Baire topology, that is, the countable product of the discrete topology on $\mathbb{N}$. For $\sigma \in \mathbb{N}^{<\mathbb{N}}$, let $[\sigma]$ be the clopen set generated by σ, i.e., $[\sigma] = \{x \in \mathbb{N}^{\mathbb{N}} : \sigma \prec x\}$. For $e \in \mathbb{N}$, let φ_e be the eth partial computable function on $\mathbb{N}$, and φ_e^α be the eth partial computable function relative to an oracle α. For a partial function φ, we write $\varphi(n) \downarrow$ if $\varphi(n)$ is defined, and $\varphi(n) \uparrow$ if $\varphi(n)$ is undefined.

As usual, for $n \in \mathbb{N}$, we often use n to denote $\{0, 1, \ldots, n-1\}$. We denote a partial function from X to Y as $f \colon \subseteq X \to Y$. We use the symbol $\mathcal{P}(Y)$ to denote the power set of a set Y. In this article, a partial function $f \colon \subseteq X \to \mathcal{P}(Y)$ is often called a *partial multi-valued function* (abbreviated as a *multifunction*), and written as $f \colon \subseteq X \rightrightarrows Y$. In computable mathematics, we often view a $\forall\exists$-formula S as a partial multifunction. Informally speaking, a (possibly false) statement $S \equiv \forall x \in X \ [Q(x) \to \exists y P(x,y)]$ is transformed into a partial multifunction $f_S \colon \subseteq X \rightrightarrows Y$ such that $\mathrm{dom}(f_S) = \{x \in X : Q(x)\}$ and $f_S(x) = \{y \in Y : P(x,y)\}$. Here, we consider formulas as partial multifunctions rather than relations in order to distinguish a *hardest* instance $f_S(x) = \emptyset$ (corresponding to a *false sentence*) and an *easiest* instance $x \in X \setminus \mathrm{dom}(f_S)$ (corresponding to a *vacuous truth*). In this sense, a relation does not correspond to a partial multifunction. More to the point, even if we exclude the hardest instances, the category of partial multifunctions and that of relations have different morphism compositions, so they have very different structures; see [35].

2. Generalized Turing reducibility

2.1. **Perfect information game.** The notion of relative computation (or Turing reducibility) has been first introduced by Turing in 1939. From that time to the present, its structure has been investigated to an extremely deep level. As a result, a vast amount of research results are known (see e.g. [14, 20, 43] for the tip of the iceberg). Traditionally, Turing reducibility is usually considered for sets $A \subseteq \mathbb{N}$ or total functions $f \colon \mathbb{N} \to \mathbb{N}$. However, a slight extension of this, the notion of Turing reducibility for partial functions $f \colon \subseteq \mathbb{N} \to \mathbb{N}$, has also been considered, and the induced structure is known to be isomorphic to the enumeration degrees; see [11, Section 11.3]. The concept of relative computability can be extended to even larger classes of functions. As one such class, we first deal with partial multi-valued functions (abbreviated as multifunctions) on $\mathbb{N}$. In recent years, the notion of partial multifunction has received a great deal of attention in computability theory and related fields; see e.g. [6].

Relative Computation Model: Let us introduce the notion of computation relative to a partial multifunction on $\mathbb{N}$. Our computation model is the same as that of an ordinary programming language, except that a program P can contain a special instruction of the form b := [?](a). The computation model accepts a number n and a partial multifunction f as inputs. The instruction b := [?](a) assigns one of the values of $f(a)$ to the variable b. So far, it is exactly the same as an oracle Turing machine. However, if $f(a)$ is undefined, the computation will never terminate. Moreover, if f is multi-valued,

i.e., if there are more than one possible values for the output of $f(\mathtt{a})$, this generally produces a nondeterministic computation.

We write $\mathtt{P}^f$ for the partial multifunction defined by the above relative computation. To be precise, for an input n, if the program $\mathtt{P}$ terminates along any path of nondeterministic computation, we declare that n is contained in the domain of $\mathtt{P}^f$, i.e., $\mathtt{P}^f(n)\downarrow$. Furthermore, if the program $\mathtt{P}$ along some path of nondeterministic computation returns m, then we declare $m \in \mathtt{P}^f(n)$.

Definition 2.1. We say that g *is Turing reducible to* f (written $g \leq_T f$) if there exists a program $\mathtt{P}$ such that $\mathtt{P}^f$ refines g. Here, for partial multifunctions g and h, we say that h refines g if, for any n, $n \in \mathrm{dom}(g)$ implies $n \in \mathrm{dom}(h)$ and $h(n) \subseteq g(n)$.

This notion coincides with ordinary Turing reducibility when restricted to total single-valued functions. One may think that this programming definition is too vague, so we give a mathematically rigorous description of this. Formally, the process of Turing reduction for partial multifunctions can also be described as a perfect information two-player game. However, since the players' abilities are asymmetric, we will describe it as a game between $\mathtt{Merlin}$ and $\mathtt{Arthur}$.

Definition 2.2 (Perfect information game). For partial multifunctions $f, g \colon \subseteq \mathbb{N} \rightrightarrows \mathbb{N}$, let us consider the following perfect information two-player game $\mathfrak{G}(f,g)$:

$$
\begin{array}{llllll}
\mathtt{Merlin:} & x_0 & & x_1 & & x_2 & & \cdots \\
\mathtt{Arthur:} & & y_0 & & y_1 & & y_2 & \cdots
\end{array}
$$

Game rules: Each player chooses a natural number at each round. Here, $\mathtt{Merlin}$ and $\mathtt{Arthur}$ need to obey the following rules.

- First, $\mathtt{Merlin}$ chooses $x_0 \in \mathrm{dom}(f)$.
- At the nth round, $\mathtt{Arthur}$ reacts with $y_n = \langle j, u_n \rangle$.
  - The choice $j = 0$ indicates that $\mathtt{Arthur}$ makes a new query u_n to g. In this case, we require $u_n \in \mathrm{dom}(g)$.
  - The choice $j = 1$ indicates that $\mathtt{Arthur}$ declares termination of the game with u_n.
- At the $(n+1)$th round, $\mathtt{Merlin}$ responds to the query made by $\mathtt{Arthur}$ at the previous stage. This means that $x_{n+1} \in g(u_n)$.

Then, $\mathtt{Arthur}$ *wins the game* $\mathfrak{G}(f,g)$ if either $\mathtt{Merlin}$ violates the rule before $\mathtt{Arthur}$ violates the rule or $\mathtt{Arthur}$ obeys the rule and declares termination with $u_n \in f(x_0)$.

Strategies: Hereafter, we require that $\mathtt{Arthur}$'s moves are chosen in a computable manner. In other words, $\mathtt{Arthur}$'s strategy is a code τ of a partial *computable* function $h_\tau \colon \subseteq \mathbb{N}^{<\mathbb{N}} \to \mathbb{N}$. On the other hand, $\mathtt{Merlin}$'s strategy is any partial function $\sigma \colon \subseteq \mathbb{N}^{<\mathbb{N}} \to \mathbb{N}$ (which is not necessarily computable). $\mathtt{Arthur}$'s strategy τ is *winning* if, as long as $\mathtt{Arthur}$ follows the strategy τ, $\mathtt{Arthur}$ wins the game, no matter what $\mathtt{Merlin}$'s strategy σ is.

Observation 2.3. *Let f and g be partial multifunctions. Then, f is Turing reducible to g if and only if $\mathtt{Arthur}$ has a computable winning strategy for $\mathfrak{G}(f,g)$.*

Remark. A similar notion for partial multifunctions on $\mathbb{N}^{\mathbb{N}}$ has been extensively studied, e.g. in [21, 32, 18, 47, 23, 17], and is known as *generalized Weihrauch reducibility*.

Indeed, Turing reducibility in the above sense is exactly the restriction of generalized Weihrauch reducibility to functions on $\mathbb{N}$.

`Arthur`'s winning strategy τ is a *one-query strategy* if, for any play following τ, either `Merlin` violates the rule or `Arthur`'s second move y_1 is of the form $\langle 1, u \rangle$, i.e., `Arthur` declares termination at the second round.

Definition 2.4. Let f and g be partial multifunctions. We say that f *is one-query Turing reducible to* g (written $f \leq^1_{LT} g$) if there exists `Arthur`'s one-query winning strategy τ for $\mathfrak{G}(f, g)$.

Equivalently, f is a one-query Turing reducible to g if and only if there exist computable functions H and K such that for any n and m,

$$m \in g(H(n)) \implies K(n, m) \in f(n).$$

Such an H is called an *inner reduction*, and K is called an *outer reduction*.

Remark. A similar notion for partial multifunctions on $\mathbb{N}^{\mathbb{N}}$ has been extensively studied under the name *Weihrauch reducibility*; see e.g. Brattka-Gherardi-Pauly [6]. Indeed, one-query Turing reducibility is exactly the restriction of Weihrauch reducibility to functions on $\mathbb{N}$. This reducibility is also called many-one reducibility in [35].

Remark. As is well known, it is very difficult to find a natural computably enumerable (c.e.) set whose Turing degree lies strictly between computable ones and the halting problem; see e.g. [30]. As one way to solve this problem of the lack of natural intermediate c.e. degrees, Simpson [41] proposed to study the Muchnik degrees of Π^0_1 subsets of Cantor space. Here, however, we present an alternative solution, which is to consider the Turing degrees of multifunctions on $\mathbb{N}$. Observe that the Turing degree of a c.e. set $A \subseteq \mathbb{N}$ is determined by its enumeration time function η_A, where $\eta_A(n)$ is the stage when n is enumerated into A if such a stage exists; otherwise $\eta_A(n) = 0$. One can easily see that the graph of η_A is always co-c.e., i.e., Π^0_1. In this light, one can consider that the counterpart of the Turing degrees of c.e. sets in the multi-valued context is the Turing degrees of multifunctions with Π^0_1 graphs. In Example 2.5, we give a natural intermediate Π^0_1 degree between computable problems and the halting problem.

To point out the relevance of the Π^0_1 multifunctions on $\mathbb{N}$ to the Π^0_1 subsets of $\mathbb{N}^{\mathbb{N}}$, note that if $f \colon \mathbb{N} \rightrightarrows \mathbb{N}$ is a Π^0_1 multifunction, then the product $\prod_{n \in \mathbb{N}} f(n)$ forms a Π^0_1 subset of $\mathbb{N}^{\mathbb{N}}$. However, be careful about that Turing reducibility for multifunctions is entirely different from Muchnik reducibility for their product sets.

Example 2.5 (Intermediate Turing degree). The following is the $\mathbb{N}$-version of a well-studied principle, called the *lesser limited principle of omniscience*.

$$\mathrm{dom}(\texttt{LLPO}) = \{e \in \mathbb{N} : |\{j < 2 : \varphi_e(j) \downarrow\}| \leq 1\},$$
$$\texttt{LLPO}(e) = \{0, 1\} \setminus \{j < 2 : \varphi_e(j) \downarrow\}.$$

There are a huge number of mathematical principles which are equivalent to `LLPO`; see Diener [12], Diener-Ishihara [13] and also Brattka-Gherardi-Pauly [6]. The principle `LLPO` may also be called *de Morgan's law for Σ^0_1 formulas*. In the realizability context, this is closely related to Lifschitz realizability [27, 45]. It is not hard to see that the Turing degree of `LLPO` strictly lies between the computable problems and the halting

problem. This also follows from our results in later sections (see Propositions 4.2 and 4.4).

Note that $\texttt{LLPO}$ is one-query Turing equivalent to a multifunction with a Π_1^0 graph. Given $e \in \mathbb{N}$, define ψ_e as follows:

$$\psi_e(0) \downarrow \iff (\exists s \in \mathbb{N}) \, [\varphi_e(0)[s] \downarrow \, \wedge \, (\forall t < s) \, \varphi_e(1)[t] \uparrow],$$
$$\psi_e(1) \downarrow \iff (\exists s \in \mathbb{N}) \, [\varphi_e(1)[s] \downarrow \, \wedge \, \varphi_e(0)[s] \uparrow],$$

where $\varphi_e(j)[s]$ is the stage s approximation of $\varphi_e(j)$. Note that it is not possible for both $\psi_e(0)$ and $\psi_e(1)$ to terminate; that is, we always have $|\{j < 2 : \psi_e(j) \downarrow\}| \leq 1$. Then we define $\texttt{L}(e) = \{0, 1\} \setminus \{j < 2 : \psi_e(j) \downarrow\}$. Obviously, the graph of the multifunction $\texttt{L} \colon \mathbb{N} \rightrightarrows \mathbb{N}$ is Π_1^0, and $\texttt{L} \equiv_{LT}^1 \texttt{LLPO}$.

2.2. **Imperfect information game.** There are various forms of computation, one of which is the notion of probabilistic computation. As a simple example, let us consider the situation where a program $\texttt{P}$ is given an oracle α at random, and for an input n, the oracle computation $\texttt{P}^\alpha(n)$ halts with probability at least $1 - \varepsilon$. In other words, this is the situation where

$$\mu(A) \geq 1 - \varepsilon \, \wedge \, (\forall \alpha \in A) \, \texttt{P}^\alpha(n) \downarrow,$$

for some set $A \subseteq 2^\mathbb{N}$. Here, μ is the uniform probability measure on $2^\mathbb{N}$ (i.e., the probability measure by infinite fair coin flips). This probabilistic computation yields a multifunction such that the value $\texttt{P}^\alpha(n)$ for each $\alpha \in A$ is a possible output. This computation has two parameters, n and A. Of course, n is an input given by us, while A is a witness that the computation halts except for probability at most ε. It is only guaranteed that such an A exists mathematically, but the computer does not know what exactly A is.

Let us write $\texttt{ProbError}_\varepsilon \texttt{P}$ for the procedure of giving an oracle randomly to the program $\texttt{P}$ and having it perform a computation with error probability at most ε. If one wants to make explicit a parameter A which witnesses that the computation succeeds with error probability at most ε for an input n, we use the following notation:

$$\texttt{ProbError}_\varepsilon \texttt{P}(n \mid A)$$

A pair $(n \mid A)$ of parameters is properly accepted only if A witness that $\texttt{P}^\alpha(n)$ halts except for probability at most ε:

$$\texttt{ProbError}_\varepsilon \texttt{P}(n \mid A) \downarrow \iff A \subseteq 2^\mathbb{N} \, \wedge \, \mu(A) \geq 1 - \varepsilon \, \wedge \, (\forall \alpha \in A) \, \texttt{P}^\alpha(n) \downarrow.$$

Then, the value $\texttt{P}^\alpha(n)$ for each $\alpha \in A$ is a possible output:

$$y \in \texttt{ProbError}_\varepsilon \texttt{P}(n \mid A) \iff \exists \alpha \in A \, [\texttt{P}^\alpha(n) = y].$$

Although the roles of n and A are entirely different, if we just treat them formally, the above process can be regarded as a partial multifunction

$$\texttt{ProbError}_\varepsilon \texttt{P} \colon \subseteq \mathbb{N} \times \mathcal{P}(2^\mathbb{N}) \rightrightarrows \mathbb{N}.$$

In this sense, both n and A can be thought of as inputs for the above multifunction, but n is an input that is disclosed during the computation, while A is an unknown input that cannot be accessed during the computation. Hence, we call n a public input, and A a secret input.

Definition 2.6. A partial multifunction $g \colon \subseteq \mathbb{N} \times \Lambda \rightrightarrows \mathbb{N}$, where Λ is a set, is called a *bilayer function* in this article (any suggestions for a better name for this notion would be welcome). In this context, a pair $(n, c) \in \mathbb{N} \times \Lambda$ is always written as $(n \mid c)$. For $(n \mid c) \in \mathrm{dom}(g)$, we call n a *public input* and c a *secret input.* Then, the public domain $\mathrm{dom}_{\mathrm{pub}}(g)$ of g is defined as the set of all $n \in \mathbb{N}$ such that $(n \mid c) \in \mathrm{dom}(g)$ for some $c \in \Lambda$.

Example 2.7. Any partial multifunction $g \colon \subseteq \mathbb{N} \rightrightarrows \mathbb{N}$ can be identified with the following bilayer function $\hat{g} \colon \subseteq \mathbb{N} \times \{*\} \rightrightarrows \mathbb{N}$:

$$\hat{g}(n \mid *) = g(n).$$

Remark. If one wants to avoid dealing with an arbitrary set Λ, one can just consider $G(n) = \{g(n \mid c) : c \in \Lambda \text{ and } (n \mid c) \in \mathrm{dom}(g)\}$, which yields $G \colon \subseteq \mathbb{N} \to \mathcal{PP}(\mathbb{N})$. Conversely, if a partial function of the form $G \colon \subseteq \mathbb{N} \to \mathcal{PP}(\mathbb{N})$ is given, one can always assume that the elements of $G(n)$ are indexed as $G(n) = \{p_c^n : c \in \Lambda_n\}$. Then, we consider $g(n \mid c)$ to mean p_c^n, which yields a partial multifunction $g \colon \subseteq \mathbb{N} \times \Lambda \rightrightarrows \mathbb{N}$.

Indeed, previous studies, such as Lee-van Oosten [26], rather deal only with $\mathcal{PP}(\mathbb{N})$-valued functions. In the terms of Bauer [2], a $\mathcal{PP}(\mathbb{N})$-valued function is called an *extended Weihrauch degree*, and a partial multifunction as seen as an extended Weihrauch degree is called a *modest extended Weihrauch degree.* However, from the point of view of advised computation, there are advantages to the way of looking at it as in Definition 2.6.

Let us consider relative computation with a bilayer function oracle. In our computation model, a secret input for an oracle acts like an advice string in computational complexity theory. For the role of advice in computability theory, we refer the reader to Brattka-Pauly [7] and Ziegler [48]. One-query computation with advice in the context of $\mathbb{N}^{\mathbb{N}}$-computation has also been discussed there.

Example 2.8. In the context of $\mathbb{N}^{\mathbb{N}}$-computation, the bilayer function $\mathtt{Advice}_{\mathbb{N}} \colon \{*\} \times \mathbb{N} \to \mathbb{N}$ defined by $\mathtt{Advice}_{\mathbb{N}}(* \mid n) = n$ can be used to deal with *nonuniform computability* [7, 48]. However, in the context of $\mathbb{N}$-computation, $\mathtt{Advice}_{\mathbb{N}}$ is too strong and produce the $\neg\neg$-topology on the effective topos [26]. Several variants of random advice in the context of $\mathbb{N}^{\mathbb{N}}$-computation have also been studied in [7, 5].

Relative Computation Model: Our computation model deals not only with one-query relative computation, but also with many-query relative computation. During a computation with a bilayer function oracle f, when the program makes a query n to f, the advisor chooses a parameter c. However, the information of c chosen by the advisor is not given to the machine, but only the information of one of the possible values of $f(n \mid c)$ is given. If this process computes a partial multifunction g when the advisor secretly makes the best choice, then we declare that g *is Turing reducible to f in the bilayered context*, and write $g \leq_{LT} f$.

Again, one may think that this programming definition is too vague, so we give a mathematically rigorous description of this. Formally, this procedure can be understood by describing it as an imperfect information game between three players, `Merlin`, `Arthur`, and `Nimue`. The player `Merlin` makes a public input x_0 and a secret input c_0 on his first move. Here, among the moves of `Merlin`, only the secret input c_0 is

invisible to `Arthur`. All of `Nimue`'s moves are visible to `Merlin`, but not to `Arthur`, a mere human being. The players `Merlin` and `Nimue`, who are not mere humans, can see all the previous moves at each round.

Definition 2.9 (Imperfect information game). For bilayer functions f and g, let us consider the following imperfect information three-player game $\mathfrak{G}(f, g)$:

$$
\begin{array}{llllll}
\texttt{Merlin:} & (x_0 \mid c_0) & & x_1 & & x_2 & & \ldots \\
\texttt{Arthur:} & & y_0 & & y_1 & & y_2 & & \ldots \\
\texttt{Nimue:} & & & z_0 & & z_1 & & z_2 & \ldots
\end{array}
$$

Game rules: Here, the players need to obey the following rules.
- First, `Merlin` chooses a pair $(x_0 \mid c_0) \in \operatorname{dom}(f)$.
- At the nth round, `Arthur` reacts with $y_n = \langle j, u_n \rangle$.
  - The choice $j = 0$ indicates that `Arthur` makes a new query u_n to g. In this case, we require $u_n \in \operatorname{dom}_{\mathrm{pub}}(g)$.
  - The choice $j = 1$ indicates that `Arthur` declares termination of the game with u_n.
- At the nth round, `Nimue` makes an advice parameter z_n such that $(u_n \mid z_n) \in \operatorname{dom}(g)$.
- At the $(n+1)$th round, `Merlin` responds to the query made by `Arthur` and `Nimue` at the previous stage. This means that $x_{n+1} \in g(u_n \mid z_n)$.

Then, `Arthur` *and* `Nimue` *win the game* $\mathfrak{G}(f, g)$ if either `Merlin` violates the rule before `Arthur` or `Nimue` violates the rule, or both `Arthur` and `Nimue` obey the rule and `Arthur` declares termination with $u_n \in f(x_0 \mid c_0)$.

Strategies: As noted above, `Arthur` can only read the moves $x_0, x_1, x_2, \ldots$, and the other players can see all the moves. Moreover, we require that `Arthur`'s moves are chosen in a computable manner. In other words, `Arthur`'s strategy is a code τ of a partial *computable* function $h_\tau \colon \subseteq \mathbb{N}^{<\mathbb{N}} \to \mathbb{N}$, which reads `Merlin`'s moves $x_0, \ldots, x_n$ and then returns y_n. On the other hand, `Merlin` and `Nimue`'s strategies are any partial functions (which are not necessarily computable).

A pair $(\tau \mid \eta)$ of `Arthur`'s computable strategy τ and `Nimue`'s strategy η is called an `Arthur`-`Nimue` strategy. An `Arthur`-`Nimue` strategy $(\tau \mid \eta)$ is *winning* if, as long as `Arthur` and `Nimue` follow the strategy $(\tau \mid \eta)$, `Arthur` and `Nimue` win the game, no matter what `Merlin`'s strategy σ is.

We now introduce a generalization of Turing reducibility for bilayer functions.

Definition 2.10. Let f and g be bilayer functions. We say that f *is bilayered Turing reducible (or LT-reducible) to* g (written $f \leq_{LT} g$) if there exists a winning `Arthur`-`Nimue` strategy for $\mathfrak{G}(f, g)$.

Obviously, bilayered Turing reducibility for partial multifunctions (which can be viewed as bilayer functions as in Example 2.7) is the same as Turing reducibility.

Remark. The notion of an `Arthur`-`Nimue` strategy is strongly related to the notion of a *dedicated sight* in Lee-van Oosten [26, Definition 4.3]. The statement that S is a (z, θ, p)-dedicated sight roughly corresponds to that $(z \mid S)$ is a winning `Arthur`-`Nimue` strategy witnessing $\dot{p} \leq_{LT} \theta$, where $\dot{p}(* \mid *) = p$ for $p \subseteq \mathbb{N}$.

Before examining bilayered Turing reducibility, we again consider one-query reductions. A winning `Arthur-Nimue` strategy $(\tau \mid \eta)$ is a *one-query strategy* if, for any play following $(\tau \mid \eta)$, either `Merlin` violates the rule or `Arthur`'s second move y_1 is of the form $\langle 1, u \rangle$, i.e., `Arthur` declares termination at the second round.

Definition 2.11. Let f and g be bilayer functions. We say that f *is one-query bilayered Turing reducible to* g (written $f \leq^1_{LT} g$) if there exists a one-query winning `Arthur-Nimue` strategy $(\tau \mid \eta)$ for $\mathfrak{G}(f, g)$.

Equivalently, f is a one-query bilayered Turing reducible to g if and only if there exist computable functions H and K and a function L such that for any $(n \mid c)$ and m,

$$m \in g(H(n) \mid L(n, c)) \implies K(n, m) \in f(n \mid c).$$

Such an H is called an *inner reduction*, and K is called an *outer reduction*. We also call L a *secret inner reduction*.

Remark. One-query bilayered Turing reducibility for bilayer functions (extended Weihrauch degrees) is simply called *Weihrauch reducibility* in Bauer [2]. The algebraic structure of the one-query bilayered Turing degrees (the extended Weihrauch degrees) has been studied there. For this reason, it may also be natural to refer to LT-reducibility as generalized extended Weihrauch reducibility.

Proposition 2.12. $\leq^1_{LT}$ *is a preorder.*

Proof. Reflexivity is trivial. For transitivity, let $\langle H_0, K_0, L_0 \rangle$ witness $f \leq^1_{LT} g$ and $\langle H_1, K_1, L_1 \rangle$ witness $g \leq^1_{LT} h$. Then $m \in h(H_1 \circ H_0(n) \mid L_1(H_0(n), \eta_0(n, c)))$ implies $K_1(H_0(n), m) \in g(H_0(n) \mid L_0(n, c))$, which implies $K_0(n, K_1(H_0(n), m)) \in f(n \mid c)$. Hence, $H_1 \circ H_0$ is an inner reduction, $(n, m) \mapsto K_0(n, K_1(H_0(n), m))$ is an outer reduction, and $(n, c) \mapsto L_1(H_0(n), L_0(n, c))$ is a secret inner reduction witnessing $f \leq^1_{LT} h$. □

Proposition 2.13. $\leq_{LT}$ *is a preorder.*

Proof. Reflexivity is trivial. For transitivity, we only need to combine the argument in Proposition 2.12 and the proof of transitivity of generalized Weihrauch reducibility [21, Proposition 4.4]. We assume that $f \leq_{LT} g$ and $g \leq_{LT} h$. To avoid confusion, we name `Arthur_0`, `Merlin_0`, and `Nimue_0` for the players in the game $\mathfrak{G}(f, g)$, and `Arthur_1`, `Merlin_1`, and `Nimue_1` for the players in the game $\mathfrak{G}(g, h)$. Let $(\tau_i \mid \eta_i)$ be a winning `Arthur_i`-`Nimue_i` strategy for the corresponding game for each $i < 2$. In the following, we assume that `Arthur_i` and `Nimue_i` always follow their winning strategies. We construct a winning `Arthur-Nimue` strategy for $\mathfrak{G}(f, h)$.

Let $(x \mid c)$ be `Merlin`'s first move in the game $\mathfrak{G}(f, h)$. Then, consider $(x \mid c)$ as `Merlin_0`'s first move in the game $\mathfrak{G}(f, g)$ as well, and simulate a play following the `Arthur_0`-`Nimue_0` strategy $(\tau_0 \mid \eta_0)$. Along such a play, if `Arthur_0` declares termination with some u at some round, then `Arthur` also declares termination with the same value u. If `Arthur_0` and `Nimue_0` make a query $(u \mid z)$ to g at some round, then think of $(u \mid z)$ as `Merlin_1`'s first move in the game $\mathfrak{G}(g, h)$, and simulate a play following the `Arthur_1`-`Nimue_1` strategy $(\tau_1 \mid \eta_1)$. During this subplay, `Arthur` and `Nimue` simply copy the moves made by `Arthur_1` and `Nimue_1`, respectively, and play them as their own moves. Here, `Merlin` copies `Merlin_1`'s moves, except for the first move, and uses them directly

in his own moves. Since $\texttt{Arthur}_1$ and $\texttt{Nimue}_1$ follow their winning strategies, $\texttt{Arthur}_1$ declares termination with some v at some round, and moreover $v \in g(u \mid z)$. Hence, one can think of such v as $\texttt{Merlin}_0$'s response to the previous move $(u \mid z)$ by $\texttt{Arthur}_0$ and $\texttt{Nimue}_0$. This allows the game $\mathfrak{G}(f, g)$ to move on to the next round (and this can be simulated by $\texttt{Arthur}$ and $\texttt{Nimue}$, since they know the value of v). Repeating this process, since $\texttt{Arthur}_0$ and $\texttt{Nimue}_0$ follow their winning strategies, $\texttt{Arthur}_0$ declares termination with some w at some round, and moreover $w \in f(x \mid c)$. Therefore, $\texttt{Arthur}_0$ also declares termination with some $w \in f(x \mid c)$ at some round. Hence, the $\texttt{Arthur}$-$\texttt{Nimue}$ strategy described above is winning for $\mathfrak{G}(f, h)$, which concludes $f \leq_{LT} h$. $\square$

Note that the rule of the game $\mathfrak{G}(f, g)$ does not mention f except for Player I's first move. Hence, if we skip Player I's first move, we can judge if a given play follows the rule without specifying f. Such a restricted game is denoted by $\mathfrak{G}(g)$. $\texttt{Arthur}$ *and* $\texttt{Nimue}$ *win the game* $\mathfrak{G}(g)$ if either $\texttt{Merlin}$ violates the rule before $\texttt{Arthur}$ or $\texttt{Nimue}$ violates the rule, or both $\texttt{Arthur}$ and $\texttt{Nimue}$ obey the rule and $\texttt{Arthur}$ declares termination.

Definition 2.14. Given a bilayer function h, let us define the new bilayer function $h^{\supset}$ as follows: An input for $h^{\supset}$ is an $\texttt{Arthur}$-$\texttt{Nimue}$ strategy $(\tau \mid \eta)$, where $\texttt{Arthur}$'s strategy τ is a public input, and $\texttt{Nimue}$'s strategy η is a secret input.

- $h^{\supset}(\tau \mid \eta)$ is defined only if, along any play following the strategy $(\tau \mid \eta)$, $\texttt{Arthur}$ and $\texttt{Nimue}$ win the game $\mathfrak{G}(h)$ whatever $\texttt{Merlin}$'s strategy is.
- $u \in h^{\supset}(\tau \mid \eta)$ if and only if there is a play in $\mathfrak{G}(h)$ that follows the strategy $(\tau \mid \eta)$ such that $\texttt{Arthur}$ declares termination with u at some round, where all players obey the rule.

The first condition says that $(\tau \mid \eta) \in h^{\supset}$ if and only if $(\tau \mid \eta)$ is a winning $\texttt{Arthur}$-$\texttt{Nimue}$ strategy for $\mathfrak{G}(h)$ in a certain sense, and in particular, $\texttt{Arthur}$ declares termination at some round unless $\texttt{Merlin}$ violates the rule. Alternatively, $h^{\supset}$ can be thought of as a universal machine for h-relative computation.

Proposition 2.15. *For bilayer functions g and h, $g \leq_{LT} h$ if and only if $g \leq_{LT}^1 h^{\supset}$.*

Proof. ($\Rightarrow$) Let $(\tau \mid \eta)$ be a winning $\texttt{Arthur}$-$\texttt{Nimue}$ strategy witnessing $g \leq_{LT} h$. Given an input $(n \mid c)$ for g, define $\tau_n(\sigma) = \tau(n {^\frown} \sigma)$ and $\eta_{n,c}(\sigma) = \eta(\langle n, c \rangle {^\frown} \sigma)$. Any $u \in h^{\supset}(\tau_n \mid \eta_{n,c})$ corresponds to a play in $\mathfrak{G}(g, h)$ following the strategy $(\tau \mid \eta)$, where $\texttt{Merlin}$'s first move is $(n \mid c)$, and $\texttt{Arthur}$ declares termination with u. Since $(\tau \mid \eta)$ is winning in $\mathfrak{G}(g, h)$, we must have $u \in g(n \mid c)$. Thus, $n \mapsto \tau_n$ is an inner reduction, $(n, c) \mapsto \eta_{n,c}$ is a secret inner reduction, and $(n, u) \mapsto u$ is an outer reduction witnessing $g \leq_{LT}^1 h^{\supset}$.

($\Leftarrow$) Let $\langle H, K, L \rangle$ witness $g \leq_{LT}^1 h^{\supset}$. As H is an inner reduction, note that $H(n)$ is (a code of) $\texttt{Arthur}$'s strategy, and think of $H(n)(\sigma)$ as $\texttt{Arthur}$'s move after reading $\texttt{Merlin}$'s moves σ. Then, define $\tau(n {^\frown} \sigma) = H(n)(\sigma)$ if $H(n)(\sigma)$ does not declare termination, i.e., $H(n)(\sigma)$ is of the form $\langle 0, u \rangle$. If $H(n)(\sigma)$ declares termination with u, then define $\tau(n {^\frown} \sigma) = \langle 1, K(n, u) \rangle$; that is, $\tau(n {^\frown} \sigma)$ declares termination with $K(n, u)$. Clearly, τ is computable. We also define $\eta(\langle n, c \rangle {^\frown} \sigma) = L(n, c)(\sigma)$.

Assume that $\texttt{Arthur}$ and $\texttt{Nimue}$ follow the strategy $(\tau \mid \eta)$ in the game $\mathfrak{G}(g, h)$. If $\texttt{Merlin}$'s first move is $(n \mid c)$, then, by the definitions of τ and η, $\texttt{Arthur}$ and $\texttt{Nimue}$ follow the strategy $(H(n) \mid L(n, c))$ in the subgame $\mathfrak{G}(h)$ until $\texttt{Arthur}$ declares termination. Since $(H(n) \mid L(n, c)) \in \mathrm{dom}(h^{\supset})$, either $\texttt{Merlin}$ violates the rule before

`Arthur` or `Nimue` violates the rule, or all players obey the rule and `Arthur` declares termination. Assume that `Merlin` follows a strategy obeying the rule. Then, in the subgame $\mathfrak{G}(h)$, `Arthur` declares termination with u at some round, i.e., $u \in h^{\supset}(H(n) \mid L(n, c))$. Hence, by the definition of τ, in the game $\mathfrak{G}(g, h)$, `Arthur` declares termination with $K(n, u)$. As $\langle H, K, L \rangle$ are reductions witnessing $g \leq_{LT}^1 h^{\supset}$, $u \in h^{\supset}(H(n) \mid L(n, c))$ implies $K(n, u) \in g(n \mid c)$. This verifies that $(\tau \mid \eta)$ is a winning `Arthur`-`Nimue` strategy witnessing $g \leq_{LT} h$. □

Proposition 2.16. $h^{\supset\supset} \equiv_{LT}^1 h^{\supset}$.

Proof. Obviously, $h^{\supset} \leq_{LT}^1 h^{\supset\supset}$. For the other direction, by the reflexivity of $\leq_{LT}^1$, we have $h^{\supset\supset} \leq_{LT}^1 h^{\supset\supset}$, which implies that $h^{\supset\supset} \leq_T h^{\supset}$ by Proposition 2.15. Similarly, we also have $h^{\supset} \leq_{LT} h$. Since $\leq_{LT}$ is transitive by Proposition 2.13, we have $h^{\supset\supset} \leq_{LT} h$. Hence, we get $h^{\supset\supset} \leq_{LT}^1 h^{\supset}$ by Proposition 2.15. □

Remark. In the context of $\mathbb{N}^{\mathbb{N}}$-computability, the closure operator $-^{\supset}$ restricted to partial multifunctions is essentially the same as the diamond operator in [32, 47].

Remark. Several variants of Weihrauch reducibility can be explained by using bilayer functions in the context of $\mathbb{N}^{\mathbb{N}}$-computation. For instance, f is computable reducible to g in the sense of [16, 21] if and only if f is one-query bilayered Turing reducible to $(g \mid$ `Advice`$_{\mathbb{N}})$ (see Definition 4.5) if we properly extend the above notions to the context of $\mathbb{N}^{\mathbb{N}}$-computability. The notion of omniscient computable/Weihrauch reducibility [29, 17, 15] can also be explained in the bilayer context.

The bilayered Turing degrees of concrete bilayer functions are examined in Sections 4 and 5.

3. Lawvere-Tierney topology

3.1. **Realizability.** For sets $p, q \subseteq \mathcal{P}(\mathbb{N})$ we define $p \leq q$ if there exists a partial computable function $\varphi \colon \subseteq \mathbb{N} \to \mathbb{N}$ such that $n \in p$ implies $\varphi(n) \in q$. Then $(\mathcal{P}(\mathbb{N}), \leq)$ forms a Heyting algebra, where the Heyting operations are given as follows:

$$p \wedge q = \{\langle m, n \rangle : m \in p \ \wedge \ n \in q\},$$
$$p \vee q = \{\langle 0, n \rangle : n \in p\} \cup \{\langle 1, n \rangle : n \in q\},$$
$$p \to q = \{e : (\forall n \in \mathbb{N}) \, [n \in p \ \to \ \varphi_e(n) \in q]\}.$$

Now we put $\Omega = \mathcal{P}(\mathbb{N})$, and consider Ω as the set of truth values in the world of computable mathematics. Indeed, the morphism tracked by $true \colon 1 \to \Omega$, where $true(*) = \mathbb{N}$, is a subobject classifier in the effective topos. For more information on the effective topos, see [22, 25, 45].

Let A be a propositional formula, where all propositional variables belong to Ω. Then A can be thought of as an element of Ω by using Heyting operations on Ω. We say that e *realizes* A if e belongs to A under the above interpretation. We also say that e *realizes* $\forall p \, A(p)$ if $e \in A(p)$ for any $p \in \Omega$. Then, A *is realizable* if some $e \in \mathbb{N}$ realizes A.

3.2. **Lawvere-Tierney topologies.** In this section, we reveal the hidden relationship between bilayered Turing degrees and Lawvere-Tierney topologies (Definition 3.2) on the effective topos. As mentioned in Section 1.1, we regard an Lawvere-Tierney topology

on the effective topos as a kind of data that indicate how much non-computability to add to the world, and thus, a topology plays the same role as an oracle. In this regard, Hyland [22] found an embedding of the Turing degrees (of total single-valued functions on $\mathbb{N}$) into the lattice of Lawvere-Tierney topologies on the effective topos. Hyland's embedding can be extended to partial functions or even partial multifunctions on $\mathbb{N}$. By extending this further, we show that there exists an isomorphism between the bilayered Turing degrees and the lattice of Lawvere-Tierney topologies on the effective topos (Corollary 3.5). This guarantees that, in the strict sense, any topology on the effective topos can be identified with a (bilayer) oracle. Note that most of the results in Section 3.2 have almost been proven by Lee and van Oosten [26], although their language is completely different from ours, and in particular they do not give any computational interpretation of their notions.

A function $\beta\colon \Omega \to \Omega$ is said to be *computably monotone* if the following is realizable:

$$\forall p, q \ [(p \to q) \to (\beta(p) \to \beta(q))].$$

In other words, there exists e such that for any $p, q \in \Omega$ if a realizes $p \to q$ then $\varphi_e(a)$ realizes $\beta(p) \to \beta(q)$. We define a preorder on computably monotone functions on Ω as follows:

$$\alpha \leq_{\mathrm{rea}} \beta \iff \text{``}\forall p \ [\alpha(p) \to \beta(p)]\text{''} \text{ is realizable},$$

that is, there exists e such that, for any $p \in \Omega$, e realizes $\alpha(p) \to \beta(p)$. For a preorder, its quotient by the induced equivalence relation is called the poset reflection.

Theorem 3.1. *The poset reflections of the following preorders are isomorphic:*

- *The one-query bilayered Turing preorder $\leq^1_{LT}$ on bilayer functions.*
- *The preorder $\leq_{\mathrm{rea}}$ on computably monotone functions on Ω.*

Proof. Given a bilayer function g, we define a function $g^{\to}\colon \Omega \to \Omega$ as follows:

$$\langle n, e \rangle \in g^{\to}(p) \iff (n \mid c) \in \mathrm{dom}(g) \text{ and } e \text{ realizes } g(n \mid c) \to p \text{ for some } c.$$

Roughly speaking, $g^{\to}(p)$ is a problem that asks us to solve a problem p with the help of g. Of the solutions n and e to $g^{\to}(p)$, we sometimes call n an *inner reduction* and e an *outer reduction*. Indeed, if we put $\dot{p}(* \mid *) = p$, then $\langle n, e \rangle \in g^{\to}(p)$ if and only if $\langle n, e \rangle$ witnesses $\dot{p} \leq^1_{LT} g$. Note that, if θ is a bilayer function, $\theta^{\to}$ is essentially the same as G_θ under the notation in Lee-van Oosten [26, page 873]. One can easily see that $g^{\to}$ is computably monotone (see also [26]). We first show the following:

$$g \leq^1_{LT} h \iff g^{\to} \leq_{\mathrm{rea}} h^{\to}.$$

For the forward direction, assume that $\langle H, K, L \rangle$ witnesses $g \leq^1_{LT} h$, and $\langle n, e \rangle$ witnesses $\dot{p} \leq^1_{LT} g$ with some c. Then, the composition $H(n)$ of inner reductions and the composition $\varphi_e \circ K$ of outer reductions witness $\dot{p} \leq^1_{LT} h$ with $L(n, c)$. This is because for any solution $y \in h(H(n) \mid L(n, c))$ we have $K(n, y) \in g(n \mid c)$ as $\langle H, K, L \rangle$ is a reduction triple, and for any $z \in g(n \mid c)$ we have $\varphi_e(z) \in \dot{p}(* \mid *) = p$; hence $\varphi_e \circ K(n, y) \in p$. Put $n' = H(n)$ and let e'_n be an index of the computable function $y \mapsto \varphi_e \circ K(n, y)$. Then we have $\langle n', e'_n \rangle \in h^{\to}(p)$. Clearly $\langle n, e \rangle \mapsto \langle n', e'_n \rangle$ is computable and independent of p. Hence we get $g^{\to} \leq_{\mathrm{rea}} h^{\to}$.

For the backward direction, let e be a realizer for $g^{\to}(p) \to h^{\to}(p)$ for any $p \in \Omega$. Given $(n \mid c)$, let us consider $p = g(n \mid c)$. It is obvious that $\langle n, \mathrm{id} \rangle$ witnesses $\dot{p} \leq^1_{LT} g$.

Thus, $\varphi_e(n, \mathrm{id}) = \langle m_n, d_n \rangle$ witnesses $\dot{p} \leq^1_{LT} h$. In other words, $y \in h(m_n \mid c')$ implies $\varphi_{d_n}(y) \in \dot{p}(* \mid *) = g(n \mid c)$ for some c'. One can find an index d such that $\varphi_d(n, x) = \varphi_{d_n}(x)$. Then, $n \mapsto m_n$ is an inner reduction, $c \mapsto c'$ is a secret inner reduction, and φ_d is an outer reduction for $g \leq^1_{LT} h$.

To show surjectivity of $g \mapsto g^\rightarrow$ on the poset reflections, given a computably monotone function $\beta \colon \Omega \to \Omega$, define a bilayer function $\beta^\leftarrow$ as follows:

$$\mathrm{dom}(\beta^\leftarrow) = \{(n \mid c) : n \in \beta(c)\}, \qquad \beta^\leftarrow(n \mid c) = c.$$

We claim that $(\beta^\leftarrow)^\rightarrow \equiv_{\mathrm{rea}} \beta$. Note that $\beta \mapsto \beta^\leftarrow$ is essentially the same as the transformation $f \mapsto \theta$ in the proof of Lee-van Oosten [26, Theorem 2.4]. Therefore, as in the proof of [26, Theorem 2.4] (that shows $f \equiv F(f) \equiv G_\theta$ under their terminology), one can verify the claim. However, the notation in [26] is very different from ours, so for the sake of completeness, we give the details here. Define $\tilde{\beta} \colon \Omega \to \Omega$ as follows:

$$\langle n, e \rangle \in \tilde{\beta}(p) \iff (\exists q \in \Omega) \, [n \in \beta(q) \text{ and } e \text{ realizes } q \to p].$$

Clearly, we have $\beta \leq_{\mathrm{rea}} \tilde{\beta}$ by considering $q = p$. To see $\tilde{\beta} \leq_{\mathrm{rea}} \beta$, let a witness computable monotonicity of β. If $\langle n, e \rangle \in \tilde{\beta}(p)$ via q, then e realizes $q \to p$ and so $\varphi_a(e)$ realizes $\beta(q) \to \beta(p)$ by computable monotonicity. As $n \in \beta(q)$, we get $\varphi_{\varphi_a(e)}(n) \in \beta(p)$. Hence, $\tilde{\beta} \equiv_{\mathrm{rea}} \beta$. Comparing the definitions of $\beta^\leftarrow$ and $\tilde{\beta}$, one can see that $\langle n, e \rangle \in \tilde{\beta}(p)$ if and only if there exists $q \in \Omega$ such that $(n \mid q) \in \mathrm{dom}(\beta^\leftarrow)$ and e realizes $\beta^\leftarrow(n \mid q) \to p$. By the definition of $g \mapsto g^\rightarrow$, this means that $\tilde{\beta} = (\beta^\leftarrow)^\rightarrow$. Hence, $\beta \equiv_{\mathrm{rea}} \tilde{\beta} = (\beta^\leftarrow)^\rightarrow$. Thus, $g \mapsto g^\rightarrow$ is a surjective embedding of the one-query bilayered Turing degrees into the poset reflection of $\leq_{\mathrm{rea}}$ on computably monotone functions. This completes the proof. $\qquad\square$

Remark. Recall that, in [2], one-query bilayered Turing reducibility is called extended Weihrauch reducibility. Together with the result in Bauer [2] that extended Weihrauch degrees and instance degrees (over a relative partial combinatory algebra) are equivalent preorders, one can deduce that the orders in Theorem 3.1 are also isomorphic to instance degrees over Kleene's first algebra.

By the proof of Theorem 3.1, note that we also have $(g^\rightarrow)^\leftarrow \equiv^1_{LT} g$ and

$$g \leq_{\mathrm{rea}} h \iff g^\leftarrow \leq^1_{LT} h^\leftarrow.$$

We next consider the notion of Lawvere-Tierney topology (also known as local operator or geometric modality), which is, in general, defined as a certain operator on the truth-value object in a given topos. It is a generalization of Grothendieck topology, and indeed, these notions are essentially equivalent on any Grothendieck topos; see e.g. [28]. The importance of this notion is that Lawvere-Tierney topologies on a topos $\mathcal{E}$ are in one-to-one correspondence with *subtoposes* of $\mathcal{E}$. In the effective topos, this notion can be described as follows (see also [25, 26]):

Definition 3.2. A function $j \colon \Omega \to \Omega$ is a *Lawvere-Tierney topology* if all of the following are realizable

(1) $\forall p \, [p \to j(p)]$.
(2) $\forall p \, [j(p \wedge q) \leftrightarrow j(p) \wedge j(q)]$.
(3) $\forall p \, [j(j(p)) \to j(p)]$.

Recall from the proof of Theorem 3.1 that $g^{\rightarrow}(p)$ is the set of reduction pairs for $\dot{p} \leq^1_{LT} g$ (where a secret reduction is not included). Therefore, by Proposition 2.15, $g^{\supset\rightarrow}(p)$ is essentially the set of Arthur's winning strategies for $\dot{p} \leq_{LT} g$. We next see that the function $g^{\supset\rightarrow} \colon \Omega \to \Omega$ is always a Lawvere-Tierney topology.

Observation 3.3. *Let h be a bilayer function. Then, $h^{\supset\rightarrow} \colon \Omega \to \Omega$ is a Lawvere-Tierney topology.*

Proof. (1) If one can solve a problem p without any help, it is clear that one can also solve the problem p with the help of $h^{\supset}$. (2) For the backward direction, if one can solve problems p and q with the help of $h^{\supset}$, then by running these strategies in parallel, one can also solve the problem $p \wedge q$ with the help of $h^{\supset}$. The forward direction is obvious.

(3) By definition, $\langle \tau, e \rangle \in h^{\supset\rightarrow}h^{\supset\rightarrow}(p)$ if and only if e realizes $h^{\supset}(\tau \mid \eta) \to h^{\supset\rightarrow}(p)$ for some η. Thus, if $u \in h^{\supset}(\tau \mid \eta)$ and $\varphi_e(u) = \langle \tau'(u), e'(u) \rangle$ then $e'(u)$ realizes $h^{\supset}(\tau'(u) \mid \eta'(u)) \to p$ for some $\eta'(u)$. As in the proof of Proposition 2.13, we combine two games, but this time in series. On the first game $\mathfrak{G}(h)$, Arthur and Nimue follow their strategies $(\tau \mid \eta)$ with one exception: Even if the strategy τ declares termination u, then Arthur do not declare termination, but move on to the next game which is also $\mathfrak{G}(h)$. Then Arthur and Nimue next follow their strategies $(\tau'(u) \mid \eta'(u))$ with one exception: If the strategy τ declares termination v, then Arthur declares termination with $\varphi_{e'(u)}(v)$. Note that if Merlin obeys the rule, then we always have $\varphi_{e'(u)}(v) \in p$. This can be viewed as a single game which is also $\mathfrak{G}(h)$, and then we write $(\tau'' \mid \eta'')$ for the Arthur-Nimue strategy described above. It is easy to check that $(\tau'' \mid \eta'') \in \mathrm{dom}(h^{\supset})$, and if $w \in h^{\supset}(\tau'' \mid \eta'')$ then $w \in p$. Therefore, the identity map realizes $h^{\supset}(\tau'' \mid \eta'') \to p$. Hence, $\langle \tau'', \mathrm{id} \rangle \in h^{\supset\rightarrow}(p)$. Clearly, $\tau \mapsto \tau''$ is computable, and thus an index of the computable function $\langle \tau, e \rangle \mapsto \langle \tau'', \mathrm{id} \rangle$ realizes $h^{\supset\rightarrow}h^{\supset\rightarrow}(p) \to h^{\supset\rightarrow}(p)$ for all p. $\qquad \square$

For any monotone function $\beta \colon \Omega \to \Omega$, consider $L(\beta) = \beta^{\leftarrow\supset\rightarrow}$. By the above observation, $L(\beta)$ is always a topology.

Theorem 3.4 (see also [26, Proposition 1.2]). *Let $\beta \colon \Omega \to \Omega$ be a computably monotone function. Then, $L(\beta)$ is the $\leq_{\mathrm{rea}}$-least topology such that $L(\beta) \geq_{\mathrm{rea}} \beta$.*

Proof. First, since $\beta^{\leftarrow} \leq^1_{LT} \beta^{\leftarrow\supset}$, we have $\beta \equiv_{\mathrm{rea}} \beta^{\leftarrow\rightarrow} \leq_{\mathrm{rea}} \beta^{\leftarrow\supset\rightarrow} = L(\beta)$ by Theorem 3.1; that is, $\beta \leq_{\mathrm{rea}} L(\beta)$ always holds. Thus, it remains to show that $\beta \leq_{\mathrm{rea}} j$ implies $L(\beta) \leq_{\mathrm{rea}} j$ for any topology j. To prove this, as in [26, Proposition 1.2], consider the following:

$$L'(\beta)(p) := \forall q \; [[(p \to q) \wedge (\beta(q) \to q)] \to q].$$

As shown in [26, Proposition 1.2], $\beta \leq_{\mathrm{rea}} j$ implies $L'(\beta) \leq_{\mathrm{rea}} j$. Hence, it remains to show $L(\beta) \leq_{\mathrm{rea}} L'(\beta)$. Here, recall that a realizer for $L(\beta)(p) = \beta^{\leftarrow\supset\rightarrow}(p)$ is a pair $\langle \mathtt{d}, \mathtt{e} \rangle$ of an inner reduction $\mathtt{d}$ and an outer reduction $\mathtt{e}$ for $\dot{p} \leq^1_{LT} \beta^{\leftarrow\supset}$, i.e., $\mathtt{e}$ realizes $\beta^{\leftarrow\supset}(\mathtt{d} \mid c) \to p$ for some c. In this case, we must have $(\mathtt{d} \mid c) \in \mathrm{dom}(\beta^{\leftarrow\supset})$, which means that $(\mathtt{d} \mid c)$ is a winning Arthur-Nimue strategy for the game $\mathfrak{G}(\beta^{\leftarrow})$.

To compute a realizer of $L'(\beta)(p)$, assume that we are given a realizer $\mathtt{a}$ of $p \to q$ and a realizer $\mathtt{b}$ of $\beta(q) \to q$ in the premise of $L'(\beta)(p)$, which are independent of q. On some play of the game $\mathfrak{G}(\beta^{\leftarrow})$, if $(n \mid z)$ is Arthur and Nimue's queries to $\beta^{\leftarrow}$ in their moves (without declaring termination) at some round, then we have $(n \mid z) \in \mathrm{dom}(\beta^{\leftarrow})$, which means that $n \in \beta(z)$ and $\beta^{\leftarrow}(n \mid z) = z$. Since $\mathtt{b}$ realizes $\beta(z) \to z$, we have

$\mathtt{b} \cdot n \in \beta^{\leftarrow}(n \mid z)$. Hence, $\mathtt{b}$ yields $\mathtt{Merlin}$'s strategy which obeys the rule. Therefore, one can simulate one of the plays of the game $\mathfrak{G}(\beta^{\leftarrow})$ from the information in $\mathtt{d}$, $\mathtt{c}$, and $\mathtt{b}$. Since $(\mathtt{d} \mid c)$ is a winning $\mathtt{Arthur\text{-}Nimue}$ strategy, and $\mathtt{Merlin}$'s strategy $\mathtt{b}$ obeys the rule, $\mathtt{Arthur}$ declares termination at some round along this play. In particular, one can compute $\mathtt{Arthur}$'s final move in this play, which yields some $m \in \beta^{\leftarrow \circlearrowright}(\mathtt{d} \mid c)$. By applying $\mathtt{e}$ to this result, we can get a realizer for p. Furthermore, by applying $\mathtt{a}$ to this result, we get a realizer for q. This procedure yields a realizer for $L(\beta)(p) \to L'(\beta)(p)$ independent of p, and thus, $L(\beta) \to L'(\beta)$ is realizable. $\qquad\square$

As in [26], we define

$$\alpha \leq_{\mathrm{L}} \beta \iff \alpha \leq_{\mathrm{rea}} L(\beta).$$

In summary, for bilayer functions f and g, we obtain

$$f \leq_{LT} g \iff f \leq_{LT}^{1} g^{\circlearrowright} \iff f^{\to} \leq_{\mathrm{rea}} g^{\circlearrowright \to} = L(g^{\to}) \iff f^{\to} \leq_{\mathrm{L}} g^{\to}.$$

Here, the first equivalence follows from Proposition 2.15, and the second one follows from Theorem 3.1. As any computably monotone function is $\equiv_{\mathrm{rea}}$-equivalent to a function of the form $f^{\to}$, this concludes the following:

Corollary 3.5. *The poset reflections of the following preorders are isomorphic:*

- *The bilayered Turing preorder $\leq_{LT}$ on bilayer functions.*
- *The preorder $\leq_{\mathrm{L}}$ on computably monotone functions on Ω.*
- *The preorder $\leq_{\mathrm{rea}}$ on Lawvere-Tierney topologies.*

In summary, for any bilayer function g, the map $g^{\circlearrowright \to}$ which, given a problem p, returns a problem asking us to giving an $\mathtt{Arthur}$'s winning strategy for $\dot{p} \leq_{LT} g$ is always a Lawvere-Tierney topology, and conversely, every Lawvere-Tierney topology on the effective topos can be described in this way.

4. On the structures of Lawvere-Tierney topologies

4.1. **Turing degrees of choices from co-m-tons.** If the domain (for public inputs) of a bilayer function g is a singleton (i.e., an input is always of the form $(* \mid c)$), then we call g a *basic bilayer function*. In many cases, such a function is not modest, where a function is modest if it is (one-query) bilayered Turing equivalent to a partial multifunction (see also Bauer [2]). In this section, we deal with the basic (non-modest) bilayer function $\mathtt{Error}_{m/k}$ for $m < k \in \mathbb{N}$ defined by

$$\mathrm{dom}(\mathtt{Error}_{m/k}) = \{(* \mid A) : A \subseteq \{0, \ldots, k-1\} \wedge |A| = m\},$$
$$\mathtt{Error}_{m/k}(* \mid A) = \{0, \ldots, k-1\} \setminus A$$

This is a problem such that m of the k choices are wrong. In particular, one-query $\mathtt{Error}_{m/k}$-relative computation is the one in which k computations are run in parallel, m of which may be wrong. Note that the basic bilayer function $\mathtt{Error}_{m/k}$ is denoted as $\mathcal{O}_m^k$ in [26]. Lee-van Oosten [26] proposed to study the structure of $(\{\mathtt{Error}_{m/k}^{\to}\}_{m<k}, \leq_{\mathrm{rea}}, \leq_{\mathrm{L}})$. By Theorem 3.1 and Corollary 3.5, this is the same as the examining the structure of $(\{\mathtt{Error}_{m/k}\}_{m<k}, \leq_{LT}^{1}, \leq_{LT})$.

Of course, the basic bilayer function $\mathtt{Error}_{m/k}$ is closely related to the well-known notion, the lessor limited principle of omniscience (recall Example 2.5). Here, we consider its generalization:

$$\mathrm{dom}(\mathtt{LLPO}_{m/k}) = \{e \in \mathbb{N} : |\{j < k : \varphi_e(j) \downarrow\}| \le m\},$$
$$\mathtt{LLPO}_{m/k}(e) = \{0, \dots, k-1\} \setminus \{j < k : \varphi_e(j) \downarrow\}.$$

Clearly, $\mathtt{LLPO}$ introduced in Example 2.5 is equivalent to $\mathtt{LLPO}_{1/2}$. As mentioned there, it yields Lifschitz realizability. The principle $\mathtt{LLPO}_{1/\ell}$ is first introduced in Richman [38], and extensively studied in constructive mathematics and related areas. For the computability-theoretic study, see Brattka-Gherardi-Pauly [6]. Note that one can deduce several results on the structure of $(\{\mathtt{LLPO}_{m/k}\}_{m<k}, \le^1_{LT})$ from the work by Cenzer-Hinman [8].

One may relativize $\mathtt{LLPO}_{m/k}$ by replacing φ_e with φ_e^α for a given oracle $\alpha \in 2^{\mathbb{N}}$, and then the resulting function is denoted by $\mathtt{LLPO}_{m/k}^\alpha$. Recall from Example 2.7 that a partial multifunction can be thought of as a bilayer function. The following is obvious:

Observation 4.1. *For any oracle α, $\mathtt{LLPO}_{m/k}^\alpha \le^1_{LT} \mathtt{Error}_{m/k}$.*

If $\alpha = \emptyset$, we can do a little better.

Proposition 4.2. $\mathtt{LLPO}_{m/k} \le^1_{LT} \mathtt{Error}_{m/k+1}$.

Proof. We define a secret inner reduction L as follows: For any $e \in \mathrm{dom}(\mathtt{LLPO}_{m/k})$,

$$L(e) = \begin{cases} \{j < k : \varphi_e(j) \downarrow\} & \text{if } |\{j < k : \varphi_e(j) \downarrow\}| = m \\ \{j < k : \varphi_e(j) \downarrow\} \cup \{k\} & \text{if } |\{j < k : \varphi_e(j) \downarrow\}| < m \end{cases}$$

One can easily check that $(* \mid L(e))$ belongs to the domain of $\mathtt{Error}_{m/k+1}$. For an outer reduction K, define $K(e, j) = j$ for any $j < k$. To compute $K(e, k)$, wait for finding m many $j < k$ such that $\varphi_e(j) \downarrow$. If it is found at some stage, then $K(e, k)$ is defined as the least $\ell < k$ such that $j \ne \ell$ for any such $j < k$, so $K(e, k) \downarrow = \ell$ implies $\varphi_e(j) \uparrow$. Otherwise, the computation never terminates, i.e., $K(e, k) \uparrow$.

We claim that $\langle L, K \rangle$ witnesses $\mathtt{LLPO}_{m/k} \le^1_{LT} \mathtt{Error}_{m/k+1}$. Assume $a \in \mathtt{Error}_{m/k+1}(* \mid L(e))$. If $a < k$ then we have $K(e, a) = a \notin \{j < k : \varphi_e(j) \downarrow\}$; hence $K(e, a) \in \mathtt{LLPO}_{m/k}$. If $a = k$ then by our definition of $L(e)$, we must have $|\{j < k : \varphi_e(j) \downarrow\}| = m$. The computation for $K(e, a)$ eventually recognizes this fact at some stage, and this implies that $K(e, a) \notin \{j < k : \varphi_e(j) \downarrow\}$; hence $K(e, a) \in \mathtt{LLPO}_{m/k}$. □

In particular, we get $\mathtt{LLPO} \le^1_{LT} \mathtt{Error}_{1/3}$. One can easily see that the above proof indeed shows that $\mathtt{LLPO}_{m/k} \le^1_{LT} \mathtt{LLPO}_{m/k+1}^{\emptyset'}$. However, it cannot be improved any further. To prove this, we need a little preparation. We say that a tree $T \subseteq \mathbb{N}^{<\mathbb{N}}$ is *n-fat* if any node $\sigma \in T$ which is not a leaf has at least n many immediate successors. We use the following easy combinatorial fact, which is a slight modification of Cenzer-Hinman [8, Proposition 2.9] (see also Lemma 5.7 below).

Fact 4.3. *For $\ell, m \in \mathbb{N}$, let T be an $(m \cdot \ell + 1)$-fat finite tree, and L_T be the set of all leaves of T. Assume that every leaf of T_η has the same length. Then, for any function $f : L_T \to \ell$ there exists an $(m+1)$-fat tree $S \subseteq T$ such that f is constant on the leaves of S.*

Proposition 4.4. $\mathtt{LLPO}_{1/\ell} \not\leq_{LT} \mathtt{Error}_{1/\ell+2}$.

Proof. The proof is by a typical recursion trick; see [23]. Suppose for the sake of contradiction that there exists a winning $\mathtt{Arthur}$-$\mathtt{Nimue}$ strategy $(\tau \mid \eta)$ witnessing $\mathtt{LLPO}_{1/\ell} \leq_{LT} \mathtt{Error}_{1/\ell+2}$. Except for the first move e, $\mathtt{Merlin}$'s move is always a number $j < \ell + 2$, which yields the tree $(\ell + 2)^{<\mathbb{N}}$ of all possible moves by $\mathtt{Merlin}$. Fix e, and then $\mathtt{Arthur}$'s strategy τ yields a partial computable function $\Phi_\tau \colon \subseteq (\ell + 2)^{<\mathbb{N}} \to \mathbb{N}$, where $\Phi_\tau(\sigma) \downarrow = u$ if and only if, after reading $\mathtt{Merlin}$'s moves σ, $\mathtt{Arthur}$'s strategy τ declares termination with u. Moreover, as $\mathtt{Nimue}$ makes a secret input $A \subseteq (\ell + 2)$ for $\mathtt{Error}_{1/\ell+2}$ at each round, $\mathtt{Nimue}$'s strategy η restricts $\mathtt{Merlin}$'s possible moves to an $(\ell + 1)$-fat finite subtree T_η of $(\ell + 2)^{<\mathbb{N}}$, where after $\mathtt{Arthur}$ declares termination, $\mathtt{Nimue}$ makes no further moves; hence if $\sigma \in T_\eta$ and $\Phi_\tau(\sigma) \downarrow$ then σ has to be a leaf of T_η. On the other hand, if $\mathtt{Arthur}$ does not declare termination, then $\mathtt{Nimue}$ makes the next move, so Φ_τ restricted to the leaves L_η of T_η yields a total function from L_η to ℓ. One can assume that every leaf of T_η has the same length: Otherwise, let t be the length of a longest node of T_η, and for each leaf ρ of T_η of length $s < t$, place a full $(\ell + 1)$-branching tree of height $t - s$ on the leaf ρ. Then, for a leaf ρ of the resulting tree T_η^*, define $\Phi_\tau^*(\rho)$ as the value $\Phi_\tau(\rho^*)$ for the unique initial segment ρ^* of ρ such that $\rho^* \in L_\eta$. Then replace T_η with T_η^* if necessary.

Since Φ_τ is ℓ-valued, and T_η is $(\ell + 1)$-fat, by Fact 4.3, there exists a 2-fat subtree S of T_η of the same height such that Φ_τ is constant on the leaves of S. Recall that Φ_τ depends on e, so there exists a computable function d such that $\Phi_\tau = \varphi_{d(e)}$. Now, we construct an algorithm $r(e)$ as follows: By brute-force, $\mathtt{Merlin}$ searches for a 2-fat finite subtree S of $(\ell + 2)^{<\mathbb{N}}$ such that $\varphi_{d(e)}$ is total and constant on the leaves of S. Let $j < \ell$ be the unique value of $\varphi_{d(e)}$ on the leaves of S. Then, we declare that $\varphi_{r(e)}(j)$ halts, and $\varphi_{r(e)}(k)$ never halts for $k \neq j$.

Since $\mathtt{Nimue}$'s move reduces the number of possible moves for $\mathtt{Merlin}$ by at most one at each round, and S is 2-fat, S contains $\mathtt{Merlin}$'s correct moves in a play following $(\tau \mid \eta)$ as a path $\rho \in S$. Since $(\tau \mid \eta)$ is winning, $\varphi_{d(e)}(\rho) = \Phi_\tau(\rho) = j \in \mathtt{LLPO}_{1/\ell}(e)$ since e is $\mathtt{Merlin}$'s first move. This means that $\varphi_e(j) \uparrow$. However, by the Kleene recursion theorem (see e.g. [11, Theorem 4.1.1] or [33, Theorem II.2.10]), there exists e such that $\varphi_e = \varphi_{r(e)}$, and by our definition, we have $\varphi_{r(e)} \downarrow$. This is a contradiction. $\quad\square$

This also shows that $\mathtt{LLPO}_{1/\ell}^{\emptyset'} \not\leq_{LT} \mathtt{Error}_{1/\ell+1}$.

Definition 4.5. For a partial multifunction f and a basic bilayer function g, we define a bilayer function $(f \mid g)$ as follows:

$$(f \mid g)(n \mid c) = f(n) \times g(* \mid c).$$

A similar argument as above also shows the following:

Proposition 4.6. $\mathtt{Error}_{1/\ell} \not\leq_{LT} (f \mid \mathtt{Error}_{1/\ell+1})$ *for any partial multifunction* f.

Proof. Note that any partial multifunction f is one-query Turing reducible to a single-valued function. This is because any choice function for f refines f. Thus, without loss of generality, one can assume that f is a single-valued function.

Suppose for the sake of contradiction that there exists there exists a winning $\mathtt{Arthur}$-$\mathtt{Nimue}$ strategy $(\tau \mid \eta)$. Except for the first move c, the second coordinate of $\mathtt{Merlin}$'s

move is always a number $j < \ell + 1$, which yields the tree $(\ell+1)^{<\mathbb{N}}$ of all possible moves. Given Arthur's moves, the first coordinates of the corresponding moves by Merlin are computable in f. Therefore, Arthur's strategy τ and the corresponding responses by Merlin yield a partial f-computable function $\Phi_\tau^f :\subseteq (\ell+1)^{<\mathbb{N}} \to \mathbb{N}$, where $\Phi_\tau^f(\sigma) \downarrow = u$ if and only if, after reading Merlin's moves whose second coordinates are σ, Arthur's strategy τ declares termination with u. Moreover, Nimue's strategy η restricts second coordinates of Merlin's possible moves to an finite subtree T_η of $(\ell+1)^{<\mathbb{N}}$. As in the proof of Proposition 4.4, one can assume that every leaf of T_η has the same length t. Then, Φ_τ^f restricted to $T_\eta \cap (\ell+1)^t$ yields a total function from $T_\eta \cap (\ell+1)^t$ to ℓ. By considering $\min\{\Phi_\tau^f(\sigma), \ell-1\}$, one can assume that Φ_τ^f is a partial ℓ-valued function even if we consider an input $\sigma \notin T_\eta \cap (\ell+1)^t$. By Fact 4.3 applied to any totalization of Φ_τ^f, one can see that there exists a 2-fat subtree S of $(\ell+1)^t$ of the same height such that Φ_τ^f takes at most one value (or undefined) on the leaves of S (where Φ_τ^f can be partial).

Let $j < \ell$ be the only possible value of Φ_τ^f on the leaves of S. Note that this value only depends on Arthur's strategy τ; hence it is independent of Merlin's first move c. Since Nimue's move reduces the number of possible moves for Merlin by at most one at each round, and S is 2-fat, S contains the second coordinates of Merlin's correct moves in a play following $(\tau \mid \eta)$ as a path $\rho \in S$. Since $(\tau \mid \eta)$ is winning, $\Phi_\tau^f(\rho)$ is defined, and j is the only possible value. Hence $j \in \mathrm{Error}_{1/\ell}(c)$ since c is Merlin's first move. However, this value j is independent of c, which is a contradiction. $\square$

One can apply combinatorial techniques developed in [8] to prove the following:

Theorem 4.7. $\mathrm{Error}_{m/k} \equiv_{LT} \mathrm{Error}_{1/\ell}$, where $\ell = \lceil \frac{k}{m} \rceil$.

This solves all problems asked in Lee-van Oosten [26, Open problems in pages 876–877].

Proof of Theorem 4.7. As in Cenzer-Hinman [8, Proposition 2.4], one can easily see that $\lceil \frac{k}{m} \rceil \leq \ell$ implies $\mathrm{Error}_{1/\ell} \leq_{LT}^1 \mathrm{Error}_{m/k}$. Now, let us think of $\mathrm{Error}_{m/k}$ as a problem of choosing a surviving block, where there are k blocks and one may secretly destroy at most m of them. As a variant of this problem, consider $\mathrm{Error}_{m/k;n}$, where there are k blocks as above, but n of them are hard blocks. One can secretly hit blocks m times, and while a normal block will break in one hit, a hard block will only break if we hit it all m times. The information about which blocks are hard is given as a public input $\langle a_i \rangle_{i<n}$, and the information on which blocks to hit is given as a secret input $\langle c_j \rangle_{j<m}$, where $a_i, c_j < k$ for any $i < n$ and $j < m$. Formally,

$$a \notin \mathrm{Error}_{m/k;n}(\langle a_i \rangle_{i<n} \mid \langle c_j \rangle_{j<m}) \iff \begin{cases} a \in \{c_j\}_{j<m} & \text{if } a \notin \{a_i\}_{i<n}, \\ \{a\} = \{c_j\}_{j<m} & \text{if } a \in \{a_i\}_{i<n}. \end{cases}$$

This notion is an analogue of a sequence of type (m, n) in [8, Theorem 2.6]. Now assume that there are k blocks, of which n are hard blocks. Consider an operation of consolidating m of the normal blocks into a single hard block. Then we now have $k - m + 1$ blocks, since we have consolidated m blocks into one. The number of normal blocks remaining is $\ell = k - n - m$, and the number of hard blocks is $n + 1$. One can secretly hit a block m times, but assuming that a hard block will always break if one

hits it all the times does not change the difficulty of the problem, so one can assume that the number of times we hit it is $m^* = \min\{\ell, m\}$. This consolidating operation transforms an instance of $\mathtt{Error}_{m/k;n}$ into an instance of $\mathtt{Error}_{m^*/k-m+1;n+1}$.

The join $f \sqcup g$ of bilayer functions f and g is defined as follows:

$$\begin{cases} (f \sqcup g)(\langle 0, n \rangle \mid c) = f(n \mid c), \\ (f \sqcup g)(\langle 1, n \rangle \mid c) = g(n \mid c). \end{cases}$$

The next claim corresponds to the formula (6) in Cenzer-Hinman [8, Theorem 2.6].

Claim. $\mathtt{Error}_{m/k;n} \leq_{LT} \mathtt{Error}_{m^*/k-m+1;n+1} \sqcup \mathtt{Error}_{m-1/k;n}$.

Proof. Assume that an input $(\langle a_i \rangle_{i<n} \mid \langle c_j \rangle_{j<m})$ for $\mathtt{Error}_{m/k;n}$ is given. There are only a finite number of patterns of consolidating m normal blocks, and the location information of the n hard blocks is given as a public input. Therefore, by brute-force, $\mathtt{Arthur}$ can try all patterns of consolidating m normal blocks. If the number of patterns is s, at the first s rounds, $\mathtt{Arthur}$ and $\mathtt{Nimue}$ make queries to $\mathtt{Error}_{m^*/k-m+1;n+1}$. For such a round, $\mathtt{Arthur}$ chooses m normal blocks $\langle b_i \rangle_{i<m}$, i.e., $\{b_i\}_{i<m} \cap \{a_i\}_{i<n} = \emptyset$, and consolidate them into a single hard block b. Then $\mathtt{Nimue}$ hits the new blocks according to the original secret input $\langle c_j \rangle_{j<m}$; that is, if $c_j \notin \{b_i\}_{i<m}$ then $c'_j = c_j$, and if $c_j \in \{b_i\}_{i<m}$ then $c'_j = b$. Then, $\mathtt{Nimue}$'s next move is given by $\langle c'_j \rangle_{j<m}$.

If $\mathtt{Merlin}$'s response u is not b at some round, then u is a solution to the original problem, so $\mathtt{Arthur}$ declares termination with u. If $\mathtt{Merlin}$'s response u is the new consolidated hard block b at each round, then the original secret input $\langle c_j \rangle_{j<m}$ does not hit m normal blocks. This is because if $\langle c_j \rangle_{j<m}$ hits m normal blocks then $\mathtt{Arthur}$ consolidates these m blocks into a single hard block b at some round, and so $\mathtt{Nimue}$ hits the new hard block m times. This means that $\mathtt{Nimue}$ breaks b, so b is not acceptable as $\mathtt{Merlin}$'s response. Hence, the original secret input $\langle c_j \rangle_{j<m}$ hits at most $m-1$ normal blocks, so $(\langle a_i \rangle_{i<n} \mid \langle c_j \rangle_{j<m})$ can also be thought of as an input of $\mathtt{Error}_{m-1/k;n}$. Thus, at round $s+1$, $\mathtt{Arthur}$ use $\langle a_i \rangle_{i<n}$ and $\mathtt{Nimue}$ use $\langle c_j \rangle_{j<m}$ as a query to $\mathtt{Error}_{m-1/k;n}$, and then $\mathtt{Merlin}$'s response u must be a solution to $(\langle a_i \rangle_{i<n} \mid \langle c_j \rangle_{j<m})$. Then, $\mathtt{Arthur}$ declares termination with u. This is a winning $\mathtt{Arthur}$-$\mathtt{Nimue}$ strategy witnessing $\mathtt{Error}_{m/k;n} \leq_{LT} \mathtt{Error}_{m^*/k-m+1;n+1} \sqcup \mathtt{Error}_{m-1/k;n}$. $\square$

The next claim corresponds to the formula (11) in Cenzer-Hinman [8, Theorem 2.6].

Claim. If $q = \lceil \frac{k-n}{m} \rceil + n$ then $\mathtt{Error}_{m/k;n} \leq_{LT} \mathtt{Error}_{1/q}$.

Proof. If $m = 1$ then $q = k$ and there is no difference between normal blocks and hard blocks; hence $\mathtt{Error}_{1/k;n} \leq_{LT} \mathtt{Error}_{1/k}$. We prove the claim by induction on m and k. By the induction hypothesis, if we put $q_0 = \lceil \frac{(k-m+1)-(n+1)}{m^*} \rceil + n + 1$ and $q_1 = \lceil \frac{k-n}{m-1} \rceil + n$, then

$$\mathtt{Error}_{m^*/k-m+1;n+1} \leq_{LT} \mathtt{Error}_{1/q_0}$$
$$\mathtt{Error}_{m-1/k;n} \leq_{LT} \mathtt{Error}_{1/q_1}$$

We clearly have $q_1 \geq q$, and moreover

$$q_0 = \left\lceil \frac{(k-m+1)-(n+1)}{m^*} \right\rceil + n + 1 \geq \left\lceil \frac{(k-m-n)}{m} \right\rceil + n + 1 = \left\lceil \frac{(k-n)}{m} \right\rceil + n = q.$$

Thus, $q_0, q_1 \geq q$ and this implies that $\mathtt{Error}_{1/q_0}, \mathtt{Error}_{1/q_1} \leq_{LT}^1 \mathtt{Error}_{1/q}$. Hence, by the previous claim,

$$\mathtt{Error}_{m/k;n} \leq_{LT} \mathtt{Error}_{m^*/k-m+1;n+1} \sqcup \mathtt{Error}_{m-1/k;n}$$
$$\leq_{LT} \mathtt{Error}_{1/q_0} \sqcup \mathtt{Error}_{1/q_1} \leq_{LT} \mathtt{Error}_{1/q} \sqcup \mathtt{Error}_{1/q} \equiv_{LT} \mathtt{Error}_{1/q}.$$

This verifies the claim.   □

In particular, if we put $n = 0$, then we have $\mathtt{Error}_{m/k} \leq_{LT} \mathtt{Error}_{1/\ell}$ as $\ell = \lceil \frac{k}{m} \rceil$. This concludes the proof of Theorem 4.7.   □

4.2. **Non-existence of a minimal topology.** We next consider the basic bilayer function $\mathtt{Error}_{m/\mathbb{N}}$ for $m \in \mathbb{N}$ defined by

$$\mathrm{dom}(\mathtt{Error}_{m/\mathbb{N}}) = \{(* \mid A) : A \subseteq \mathbb{N} \wedge |A| = m\},$$
$$\mathtt{Error}_{m/\mathbb{N}}(* \mid A) = \mathbb{N} \setminus A$$

In Lee-van Oosten [26, Propositions 5.1 and 5.2], it is shown that $\mathtt{Error}_{m/\mathbb{N}} <_{LT}^1 \mathtt{Error}_{m+1/\mathbb{N}}$, but $\mathtt{Error}_{1/\mathbb{N}} \equiv_{LT} \mathtt{Error}_{m/\mathbb{N}}$. Interestingly, as shown in [26, Proposition 5.5], $\mathtt{Error}_{1/\mathbb{N}}$ is the $\leq_{LT}$-least basic bilayer function which is strictly $\geq_{LT}$-above $\mathtt{Id}$. However, we include bilayer functions, $\mathtt{Error}_{1/\mathbb{N}}$ is not the $\leq_{LT}$-least one. For instance, the basic bilayer function $\mathtt{Error}_{1/\mathbb{N}}$ is closely related to the notion called *all-or-counique choice* [6]:

$$\mathrm{dom}(\mathtt{ACC}) = \mathbb{N}, \qquad \mathtt{ACC}(e) = \begin{cases} \mathbb{N} \setminus \{\varphi_e(e)\} & \text{if } \varphi_e(e) \downarrow \\ \mathbb{N} & \text{if } \varphi_e(e) \uparrow \end{cases}$$

One may relativize $\mathtt{ACC}$ by replacing φ_e with φ_e^α for a given oracle $\alpha \in 2^\mathbb{N}$, and then the resulting function is denoted by $\mathtt{ACC}^\alpha$. Again, we think of a partial multifunction as a bilayer function as in Example 2.7. The following is obvious:

Observation 4.8. $\mathtt{Id} <_{LT} \mathtt{ACC}^\alpha <_{LT} \mathtt{Error}_{1/\mathbb{N}}$ *for any oracle* α.

Lee [25, Open Problem 3.5.18] asked if there is the least topology strictly above $\mathtt{Id}$. By Corollary 3.5, it is the same as asking if there is the $\leq_{LT}$-least bilayer function which is strictly $\geq_{LT}$-above $\mathtt{Id}$. If such a function exists, then it must be a non-basic bilayer function. To solve this problem, given a partial function $g: \subseteq \mathbb{N} \to \mathbb{N}$, we consider the following multifunction $\mathtt{Avoid}_g: \mathbb{N} \rightrightarrows \mathbb{N}$:

$$\mathtt{Avoid}_g(n) = \begin{cases} \mathbb{N} \setminus \{g(n)\} & \text{if } n \in \mathrm{dom}(g) \\ \mathbb{N} & \text{otherwise.} \end{cases}$$

We first show that there exists no $\leq_T$-least partial multifunction which is strictly $\geq_T$-above $\mathtt{Id}$.

Lemma 4.9. *Let* $\mathtt{P}: \subseteq \mathbb{N} \rightrightarrows \mathbb{N}$ *be a partial multifunction such that* $\mathtt{P} >_T \mathtt{Id}$. *Then, there exists a total function* $g: \mathbb{N} \to \mathbb{N}$ *such that*

$$\mathtt{Id} <_T \mathtt{Avoid}_g \quad and \quad \mathtt{P} \not\leq_T \mathtt{Avoid}_g.$$

Proof. Let $\alpha_{\mathtt{P}} \in \mathbb{N}^{\mathbb{N}}$ be an oracle coding the full information of $\mathtt{P}$. For instance,

$$\alpha_{\mathtt{P}}(\langle n, m\rangle) = \begin{cases} 0 & \text{if } n \in \mathrm{dom}(\mathtt{P}) \ \wedge \ m \notin \mathtt{P}(n) \\ 1 & \text{if } n \in \mathrm{dom}(\mathtt{P}) \ \wedge \ m \in \mathtt{P}(n) \\ 2 & \text{if } n \notin \mathrm{dom}(\mathtt{P}) \end{cases}$$

Let $g \colon \mathbb{N} \to 2$ be an $\alpha_{\mathtt{P}}$-generic real (that is, g is contained in any dense $\alpha_{\mathtt{P}}$-computable open set in Cantor space $2^{\mathbb{N}}$, and such a g exists by the Baire category theorem). By genericity, for any computable function f, there are infinitely many n such that $f(n) = g(n)$; hence $\mathtt{Id} <_T \mathtt{Avoid}_g$. Suppose for the sake of contradiction that $\mathtt{Arthur}$ has a winning strategy Ψ for $\mathtt{P} \leq_T \mathtt{Avoid}_g$. Then we show the following claim:

Claim. For any $s \in \mathbb{N}$, there exist $n \in \mathbb{N}$ and $\sigma \in \mathbb{N}^{<\mathbb{N}}$ such that

$$\min \sigma > s \ \wedge \ \Psi(n, \sigma) \downarrow = \langle 1, y\rangle \ \wedge \ \mathtt{P}(n) \downarrow \ \wedge y \notin \mathtt{P}(n),$$

where recall that by $\Psi(n, \sigma) = \langle 1, y\rangle$ we mean that, after reading $\mathtt{Merlin}$'s moves (n, σ), $\mathtt{Arthur}$ declares termination with y.

Proof. Otherwise, there is a number s refuting the claim. Let $\Psi_1(n, \sigma)$ denote the second coordinate of $\Psi(n, \sigma)$. Let us consider a sequence $a_0, a_1, \ldots$ such that $a_{\ell+1} \in \mathtt{Avoid}_g(\Psi_1(n, a_0, a_1, \ldots, a_\ell))$ for any ℓ, i.e., $\langle a_i\rangle$ is $\mathtt{Merlin}$'s moves obeying the rule. It is clear that there always exists such an $a_\ell > s$ since $\mathtt{Avoid}_g$ only reduces the number of possible values by one. Since Ψ is winning, after reading a finite initial segment of the sequence $\langle a_i\rangle$, $\mathtt{Arthur}$ declares termination; that is, $\Psi(n, a_0, a_1, \ldots, a_\ell) = \langle 1, y\rangle$ for some ℓ and y.

In particular, there exists σ with $\min \sigma > s$ such that $\Psi(n, \sigma) \downarrow = \langle 1, y\rangle$ for some y. Given n, without knowing the information about g, one can effectively find such a σ by brute-force. Then define $f(n)$ as $\Psi_1(n, \sigma)$. Clearly, f is a computable function. However, since the claim is supposed to fail, we have either $\mathtt{P}(n) \uparrow$ or $f(n) = y \in \mathtt{P}(n)$. Hence, the computable function f refines $\mathtt{P}$, which means $\mathtt{P} \leq_T \mathtt{Id}$. However this contradicts our assumption on $\mathtt{P}$. $\square$

Given s, one can find an n and a σ in the above claim in an $\alpha_{\mathtt{P}}$-computable manner since $\alpha_{\mathtt{P}}$ contains the full information of $\mathtt{P}$. For a given t, put $s(t) = \max\{g(n) : n < t\}$, and for this $s = s(t)$, we write n_t and $\sigma_t = (a_i^t)_{i<\ell}$ for n and σ in the claim. By the above claim, after reading $\mathtt{Merlin}$'s play (n_t, σ_t), $\mathtt{Arthur}$'s strategy Ψ declares termination, but fails to compute a solution of $\mathtt{P}$. However, since Ψ is a winning strategy for $\mathtt{P} \leq_T \mathtt{Avoid}_g$, $\mathtt{Merlin}$ must have violated the rule at some round. In other words, there exists $j < \ell$ such that $a_{j+1}^t \notin \mathtt{Avoid}_g(\Psi_1(n, a_0^t, a_1^t, \ldots, a_j^t))$. Put $m_j^t = \Psi_1(n, a_0^t, a_1^t, \ldots, a_j^t)$, and we now have $a_{j+1}^t = g(m_j^t)$ for some $j < \ell$. Note that, since $a_j^t > s$ for any j, if $m_j^t < t$ then $g(m_j^t) \neq a_{j+1}^t$ by our choice of $s = s(t)$. Now consider the finite set E_t defined by

$$E_t = \{\langle m_j^t, a_{j+1}^t\rangle : j < \ell \ \wedge \ m_j^t \geq t\}.$$

Given t, one can find the canonical code of E_t is an $\alpha_{\mathtt{P}}$-computable manner. Note that there exists $\langle m, a\rangle \in E_t$ such that $g(m) = a$. However, note that

$$D = \{\tau \in 2^{<\mathbb{N}} : (\exists t \in \mathbb{N})(\forall \langle m, a\rangle \in E_t) \ \tau(m) \neq a\}$$

yields a dense $\alpha_{\mathtt{P}}$-computable open set in Cantor space $2^{\mathbb{N}}$. This is because, given a binary string τ, choose $t > |\tau|$, and extend τ to τ^* so that $\tau^*(m) \neq a$ for any $\langle m, a\rangle \in E_t$.

This is doable since $\langle m, a \rangle \in E_t$ obviously implies $m \geq t$. In this way, any τ extends to $\tau^* \in D$ and thus D is dense. Since g is α_{P}-generic, we have $g \in D$; that is, there exists t such that $g(m) \neq a$ for any $\langle m, a \rangle \in E_t$. However, by the property of E_t, for any t, we must have a pair $\langle m_j^t, a_{j+1}^t \rangle \in E_t$ such that $g(m_j^t) = a_{j+1}^t$, a contradiction. $\qquad\square$

Theorem 4.10. *There exists no $\leq_{LT}$-minimal bilayer function which is strictly $\geq_{LT}$-above* Id; *that is, for any bilayer function* P $>_{LT}$ Id *there exists a bilayer function* Q *such that* Id $<_{LT}$ Q $<_{LT}$ P. *Hence, there exists no minimal Lawvere-Tierney topology which is strictly above* Id.

Proof. Let P be a bilayer function such that P $>_{LT}$ Id. Put $\mathsf{P}_n(* \mid c) = \mathsf{P}(n \mid c)$, and then P_n is a basic bilayer function. If $\mathsf{P}_n \not\leq_{LT}$ id, then by minimality of $\mathtt{Error}_{1/\mathbb{N}}$ ([26, Proposition 5.5]), we have $\mathtt{Error}_{1/\mathbb{N}} \leq_{LT} \mathsf{P}_n \leq_{LT}$ P. By Observation 4.8, P cannot be minimal. Thus, one can assume that $\mathsf{P}_n \leq_{LT}$ Id for any $n \in \mathrm{dom}(\mathsf{P})$. This implies that, since P_n is a basic bilayer function, if $n \in \mathrm{dom}(\mathsf{P})$, then $\tilde{\mathsf{P}}(n) := \bigcap_c \mathsf{P}_n(* \mid c)$ is nonempty. This is because, at some round in the reduction game for $\mathsf{P}_n \leq_{LT}$ Id, Arthur's winning strategy declares termination with some $u \in \mathsf{P}_n(* \mid c)$, which is independent of c, as c is invisible to Arthur. Then, since $\tilde{\mathsf{P}}$ is a partial multifunction, by Lemma 4.9, there exists a total function g such that Id $<_T \mathtt{Avoid}_g$ and $\tilde{\mathsf{P}} \not\leq_T \mathtt{Avoid}_g$.

We claim that P $\not\leq_{LT} \mathtt{Avoid}_g$ also holds. Otherwise, there exists Arthur's winning strategy τ witnessing P $\leq_{LT} \mathtt{Avoid}_g$, where note that, since $\mathtt{Avoid}_g$ is a partial multifunction, Nimue does not intervene in the game. Let $(n \mid c)$ be Merlin's first move. Since τ is winning, Arthur declares termination with some $u \in \mathsf{P}(n \mid c)$. However, since c is invisible to Arthur, the last value u only depends on n. This implies that $u \in \tilde{\mathsf{P}}(n) = \bigcap_c \mathsf{P}(n \mid c)$. Namely, the strategy τ also witnesses $\tilde{\mathsf{P}} \leq_T \mathtt{Avoid}_g$. However, this contradicts our choice of g.

Moreover, since the Lawvere-Tierney topologies form a lattice, the Turing degrees of bilayer functions also form a lattice by Corollary 3.5. For an explicit description of the infimum operation, we define the meet A $\sqcap$ B of bilayer functions A and B as follows:

$$(\mathsf{A} \sqcap \mathsf{B})(m, n \mid c, d) = \big(\{0\} \times \mathsf{A}(m \mid c)\big) \cup \big(\{1\} \times \mathsf{B}(n \mid d)\big).$$

Note that this is a bilayer analogue of the meet in the Weihrauch lattice [6]. We claim that the meet Q $:=$ P $\sqcap$ $\mathtt{Avoid}_g$ of P and $\mathtt{Avoid}_g$ strictly lies between Id and P. As P $\not\leq_{LT} \mathtt{Avoid}_g$, we have Q $<_{LT}$ P. If Id $\not\leq_{LT}$ Q, then Q $\leq_{LT}$ Id, so Q is computable. This means that there exists a computable function p such that $p(n, m) \in \mathsf{Q}(n, m \mid c, *)$ for n, m, c. First consider the case that for any $n \in \mathrm{dom}(\mathsf{P})$ there exists $m \in \mathbb{N}$ such that $p(n, m)$ is of the form $\langle 0, k \rangle$. For such an m, since $\mathtt{Avoid}_g$ is total, we also have $\langle n, m \rangle \in \mathrm{dom}(\mathsf{Q})$, and thus $p(n, m) \in \mathsf{Q}(n, m \mid c, *)$, so $k \in \mathsf{P}(n \mid c)$ for any c. In this case, given $n \in \mathrm{dom}(\mathsf{P})$, by brute-force, we effectively search for $m \in \mathbb{N}$ such that the first coordinate of $p(n, m)$ is 0, then return its second coordinate, which must be a solution to $\mathsf{P}(n \mid c)$ as seen above. Hence, this procedure witnesses that P is computable, which contradicts the assumption P $>_{LT}$ Id. Thus, there must exist $n \in \mathrm{dom}(\mathsf{P})$ such that $p(n, m)$ is of the form $\langle 1, k \rangle$, for any $m \in \mathbb{N}$. As in the above argument, we have $p(n, m) \in \mathsf{Q}(n, m \mid c, *)$, so $k \in \mathtt{Avoid}_g(m)$. Then, the algorithm which, given $m \in \mathbb{N}$, returns the second coordinate of $p(n, m)$ witnesses that $\mathtt{Avoid}_g$ is computable. However, this contradicts the property $\mathtt{Avoid}_g >_{LT}$ Id. Consequently, Id $<_{LT}$ Q $<_{LT}$ P. This verifies the first assertion. Then, the second assertion follows from Corollary 3.5. $\qquad\square$

This solves Lee's problem [25, Open Problem 3.5.18].

5. Other topologies

5.1. **Probabilistic computation.** As in Section 2.2, we consider bilayer functions expressing certain kinds of probabilistic computation. However, unlike Section 2.2, for the sake of brevity in discussion, we require that a parameter A be compact. By inner regularity, every μ-measurable set in $2^{\mathbb{N}}$ is approximated from the inside by a compact set, so adding this assumption does not make much difference. Here, recall that μ is the uniform probability measure on $2^{\mathbb{N}}$. Then we consider the following bilayer function:

$$\mathtt{ProbError}_\varepsilon(\langle e, n\rangle \mid A) \downarrow \Longleftrightarrow A \subseteq 2^{\mathbb{N}} \text{ is compact}$$
$$\wedge \ \mu(A) \geq 1 - \varepsilon \ \wedge \ (\forall \alpha \in A) \ \varphi_e^\alpha(n) \downarrow .$$
$$\mathtt{ProbError}_\varepsilon(\langle e, n\rangle \mid A) = \{\varphi_e^\alpha(n) : \alpha \in A\}.$$

One would say that the subtopos obtained from the Lawvere-Tierney topology corresponding to $\mathtt{ProbError}_\varepsilon$ is the "*world of probabilistically computable mathematics with error probability ε.*" Surprisingly, we show that $\mathtt{ProbError}$ induces exactly the same topology as Lee-van Oosten's function $\mathtt{Error}$.

Proposition 5.1. *For any $p, q \in \mathbb{N}$ with $p \leq q$, $\mathtt{ProbError}_{p/q} \equiv_{LT} \mathtt{Error}_{p/q}$.*

Proof. Obviously, we have $\mathtt{Error}_{p/q} \leq_{LT} \mathtt{ProbError}_{p/q}$. We show the other direction. Let $(\langle e, n\rangle \mid A)$ be an input for $\mathtt{ProbError}_{p/q}$ and assume that $\mathtt{ProbError}_{p/q}(\langle e, n\rangle \mid A) \downarrow$. This means that $\varphi_e^\alpha(n) \downarrow$ for any $\alpha \in A$. Since A is compact, there exists t such that the stage t approximation $\varphi_{e,t}^{\alpha \restriction t}(n)$ halts for any $\alpha \in A$. Note that $\langle e, n\rangle \mapsto t$ is not necessarily computable, and thus $\mathtt{Arthur}$ does not have access to this information, but $\mathtt{Nimue}$ does have access to it.

The public input $\langle e, n\rangle$ can be thought of as a partial function $\alpha \mapsto \varphi_e^\alpha(n)$. This induces the push-forward measure on $\mathbb{N}$ defined by $\nu(j) = \mu(\{\alpha \in 2^{\mathbb{N}} : \varphi_e^\alpha(n) \downarrow = j\})$. We consider its finitary approximation; that is, put $E_j^s = \{\alpha \in 2^{\mathbb{N}} : \varphi_{e,s}^{\alpha \restriction s}(n) \downarrow = j\}$, and define $\nu_s(j) = \mu(E_j^s)$. For each stage s, the value $\nu_s(j)$ is rational, and moreover $\nu_s(j) > 0$ happens for at most finitely many $j \in \mathbb{N}$. Let $\langle j_i\rangle_{i < \ell(s)}$ be a list of all such j's. Clearly, $s \mapsto \langle \nu_s(j_i)\rangle_{i < \ell(s)}$ is computable. Since we have only finitely many rationals $\langle \nu_s(j_i)\rangle_{i < \ell(s)}$, we can assume that all values have the same denominator and are of the form $\nu_s(j_i) = a_{s,i}/qr_s$. Note that we clearly have $\sum_{i < \ell(s)} a_{s,i} \leq qr_s$. Fix a pairwise disjoint sequence $(J_i^s)_{i < \ell(s)}$ such that $J_i^s \subseteq qr_s$ and $|J_i^s| = a_{s,i}$.

Since $\mathtt{Error}_{pr_s/qr_s} \leq_{LT} \mathtt{Error}_{p/q}$ by Theorem 4.7, there is $\mathtt{Arthur}$'s winning strategy τ_s witnessing this fact. As $s \mapsto r_s$ is computable, one can easily see that the proof of Theorem 4.7 ensures that $s \mapsto \tau_s$ is also computable. Hence, instead of taking $\mathtt{Nimue}$'s moves as secret inputs to $\mathtt{Error}_{p/q}$, we can take them directly as secret inputs to $\mathtt{Error}_{pr_s/qr_s}$ through the above reduction implicitly. Here, note that such a conversion takes multiple rounds through $\mathtt{Arthur}$ and $\mathtt{Nimue}$'s moves, but we do not count this number of rounds, and without mentioning it, all conversions are assumed to be done automatically.

Thus, at the $(s+1)$-st round, after seeing $\mathtt{Nimue}$'s previous move, we assume that $\mathtt{Merlin}$ plays a move j less than qr_s. Then $\mathtt{Arthur}$ reacts to this. If $\mathtt{Merlin}$'s move j is contained in J_i^s for some $i < \ell(s)$, then $\mathtt{Arthur}$ declares termination with j_i. Otherwise,

`Arthur` declares that the game proceeds to the next round, and urges `Nimue` to make the next move. Now we describe `Nimue`'s move B_s at the s-th round. `Nimue` reads the secret input A and then define B_s as follows: For any $m \in J_i^s$, we define

$$m \in B_s \iff A \cap E_{j_i}^s = \emptyset.$$

For $m \in qr_s \setminus \bigcup_{i < \ell(s)} J_i^s$, we define

$$m \in B_s \iff A \cap E_\uparrow^s = \emptyset,$$

where $E_\uparrow^s = \{\alpha \in 2^\mathbb{N} : \varphi_{e,s}^{\alpha \restriction s}(n) \uparrow\}$. One can see that if $t \geq s$ then $qr_s \setminus \bigcup_{i < \ell(s)} J_i^s$ is included in B_s, where t is a number mentioned in the first paragraph of this proof. Note that, in order to enumerate $a_{s,i}$ many elements in J_i^s into B_s, the measure of A needs to be removed by $\nu_s(j_i) = a_{s,i}/qr_s$. Similarly, in order to enumerate $qr_s - \sum_{i < \ell_s} a_{i,s}$ elements in $qr_s \setminus \bigcup_{i < \ell(s)} J_i^s$ into B_s, the measure of A needs to be removed by

$$\mu(E_\uparrow^s) = 1 - \sum_{i < \ell(s)} \mu(E_{j_i}^s) = \frac{qr_s - \sum_{i < \ell_s} a_{i,s}}{qr_s}.$$

Since the measure of A is greater than or equal to $1 - p/q$, the measure removed from A should be at most pr_s/qr_s. Therefore, only at most pr_s many elements can be enumerated into B_s, so $(* \mid B_s)$ belongs to the domain of Error_{pr_s/qr_s}. Hence, `Nimue`'s move B_s obeys the rule.

We claim that `Arthur` and `Nimue` win this game along the play described above. Note that `Arthur` declares termination by the $(t + 1)$-th round. This is because the complement of $\bigcup_{i < \ell(s)} J_i^s$ is included in B_s, and thus `Merlin` must play a move from $\bigcup_{i < \ell(s)} J_i^s$ at the $(t+1)$-th round. In response to this move, `Arthur`'s strategy described above declares termination. We now assume that `Arthur` declares termination with j_i^s at the $(s + 1)$-st round. In order for this to happen, `Merlin`'s previous move must belong to J_i^s. In such a case, `Nimue`'s previous move B_s must satisfy $J_i^s \nsubseteq B_s$. This means that $A \cap E_{j_i}^s \neq \emptyset$. In particular, there exists $\alpha \in A$ such that $\varphi_e^\alpha(n) = j_i$. Hence, $j_i \in \text{ProbError}_{p/q}(\langle e, n \rangle \mid A)$. $\square$

One can also consider a counterpart of $\text{ProbError}_\varepsilon$ in the context of partial multi-functions, which is known as *weak weak König's lemma* [5]. Let P_e be the e-th Π_1^0 subset of $2^\mathbb{N}$ (or the set of all infinite paths though the e-th primitive recursive subtree of $2^{<\mathbb{N}}$).

$$\text{WWKL}_\varepsilon(\langle e, n, i \rangle) \downarrow \iff \mu(P_i) \geq 1 - \varepsilon \ \wedge \ (\forall \alpha \in P_i) \ \varphi_e^\alpha(n) \downarrow .$$
$$\text{WWKL}_\varepsilon(\langle e, n, i \rangle) = \{\varphi_e^\alpha(n) : \alpha \in P_i\}.$$

As in Proposition 5.1, one can show that $\text{WWKL}_{p/q} \equiv_{LT} \text{LLPO}_{p/q}$. However, of course, weak weak König's lemma is known to be much stronger than the lessor limited principle of omniscience. Indeed, if we consider analogues of `WWKL` and `LLPO` in the Kleene-Vesley algebra (i.e., in the context of $\mathbb{N}^\mathbb{N}$-computation), then one can easily see that `WWKL` is strictly above `LLPO` with respect to Turing reducibility for $\mathbb{N}^\mathbb{N}$-computability (i.e., generalized Weihrauch reducibility). Therefore, Proposition 5.1 is a phenomenon specific to the effective topos. If we consider another (relative) realizability topos, such as the Kleene-Vesley topos, the situation would be completely different.
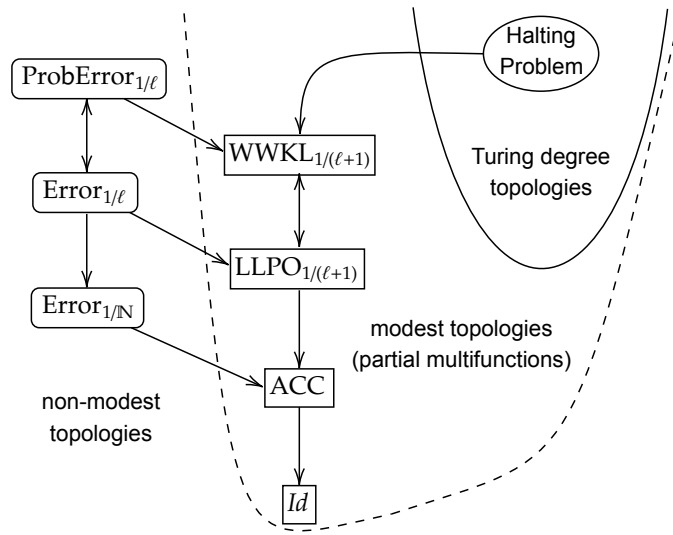
FIGURE 1. Lower parts of Lawvere-Tierney topologies on the effective topos

Figure 1 summarizes some of basic implications about the $\leq_{\mathrm{rea}}$-ordering on the Lawvere-Tierney topologies on the effective topos, where $A \rightarrow B$ means $B \leq_{LT} A$ (or $B^{\supset\rightarrow} \leq_{\mathrm{rea}} A^{\supset\rightarrow}$). By our results, there are no more implications in Figure 1.

5.2. **Cofinite choice.** The basic bilayer functions we have dealt with so far was of no help at all for computing a single-valued function when treated as an oracle. However, indeed, there is a known basic bilayer function which is rather powerful when considered as an oracle. Pitts [37, Example 5.8] introduced the following basic (non-modest) bilayer function Cofinite:

$$\mathrm{dom}(\texttt{Cofinite}) = \{*\} \times \mathbb{N}, \qquad \texttt{Cofinite}(* \mid n) = \{m \in \mathbb{N} : m \geq n\}.$$

Note that one-query Cofinite-relative computation is the one in which countably many computations are run in parallel, a finite number of which may be wrong. Pitts [37] observed that the above function yields a Lawvere-Tierney topology $\mathcal{J}$ on the effective topos such that id $<_{\mathrm{rea}} \mathcal{J} <_{\mathrm{rea}} \neg\neg$. Here, note that $\mathcal{J} = \texttt{Cofinite}^{\supset\rightarrow}$ in our terminology. Interestingly, van Oosten [46, Theorem 2.2] showed that, for a total function $f \colon \mathbb{N} \to \mathbb{N}$, the topology $\mathcal{J}$ forces f to be decidable if and only if f is hyperarithmetic. In other words, for a total function $f \colon \mathbb{N} \to \mathbb{N}$, $f \leq_{LT} \texttt{Cofinite}$ if and only if f is hyperarithmetic. For the basics of hyperarithmetic sets, we refer the reader to Sacks [40] and Chong-Yu [10]. It is also known that $\texttt{Error}_{1/\ell} \not\leq_{LT} \texttt{Cofinite}$ for any $\ell \in \mathbb{N}$ (see [26, Proposition 5.11]).

Pitts' function Cofinite also has a partial multi-valued counterpart, which has been studied as *cofinite choice* [4, 1] or *bound* [19] in the context of $\mathbb{N}^{\mathbb{N}}$-computation. We introduce cofinite choice relative to an oracle $\alpha \in 2^{\mathbb{N}}$ as follows:

$$\mathsf{C}_{\mathsf{cof}}^{\alpha}(e) \downarrow \iff |\{n \in \mathbb{N} : \varphi_e^{\alpha}(n) \downarrow\}| \text{ is finite,}$$

$$\mathsf{C}_{\mathsf{cof}}^{\alpha}(e) = \{n \in \mathbb{N} : \varphi_e^{\alpha}(n) \uparrow\}.$$

It is obvious that $\mathsf{C}_{\mathsf{cof}}^{\alpha} \leq_{LT} \texttt{Cofinite}$ for any oracle $\alpha \in 2^{\mathbb{N}}$. Moreover, we also have $\mathsf{C}_{\mathsf{cof}}^{\alpha} \leq_{T} \alpha'$, where α' is the Turing jump of α. To estimate the strength of cofinite choice,

we consider the following equivalent definition of the hyperarithmetical hierarchy based on the effective Baire hierarchy.

Definition 5.2 (Effective Baire hierarchy). For each computable ordinal ξ, we define a set B_ξ of total functions on $\mathbb{N}$ as follows: First, B_0 is the set of all total computable functions on $\mathbb{N}$, and a B_0-code of $f \in B_0$ is a program code computing f. For $\xi > 0$, B_ξ is the set of functions $f \colon \mathbb{N} \to \mathbb{N}$ such that $f = \lim_{s \to \infty} f_s$ for some sequence $(f_n)_{n \in \mathbb{N}}$, where there exists an algorithm Φ_f which, given $n \in \mathbb{N}$, returns a B_ζ-code of $f_n \in B_\zeta$ for some $\zeta < \xi$. A B_ξ-code of $f \in B_\xi$ is a pair of a code of ξ and a code of such Φ_f.

By the Shoenfield limit lemma (see e.g. [43, Lemma III.3.3] or [33, Proposition IV.1.19]), B_n corresponds to $\emptyset^{(n)}$-computability for $n \in \mathbb{N}$, and B_ξ corresponds to $\emptyset^{(\xi+1)}$-computability for an infinite ordinal ξ, where $\emptyset^{(\xi)}$ is the ξ-th Turing jump of a computable function. Obviously, Definition 5.2 of $f \in B_\xi$ produces a computable well-founded tree T_f whose leaves are labeled by computable functions. Here, such a T_f is *full-splitting*, that is, if $\sigma \in T_f$ is not a leaf, then $\sigma^\frown n \in T_f$ for any $n \in \mathbb{N}$.

Conversely, let $T \subseteq \mathbb{N}^{<\mathbb{N}}$ be a computable full-splitting well-founded tree. We inductively define $\operatorname{rank}_T(\sigma)$, the rank of σ, as follows: The rank of a leaf of T is 0, and if $\sigma \in T$ is not a leaf, then the rank of σ is $\sup\{\operatorname{rank}_T(\sigma^\frown n) + 1 : n \in \mathbb{N}\}$. Then, $\operatorname{rank}(T)$, the rank of T, is defined as the rank of its root.

Let L_T be the set of leaves of T. A *computable assignment* is a computable function $h \colon L_T \times \mathbb{N} \to \mathbb{N}$. We inductively label each node σ of T with a function $h_\sigma \colon \mathbb{N} \to \mathbb{N}$ or an undefined symbol $\uparrow$ as follows: If ρ is a leaf of T, then put $h_\rho(n) = h(\rho, n)$. If $\sigma^\frown s$ is labeled by $\uparrow$ for some $s \in \mathbb{N}$, then h_σ is also labeled by $\uparrow$. If $\lim_{s \to \infty} h_{\sigma^\frown s}(n)$ exists for all $n \in \mathbb{N}$, then σ is labeled by the pointwise limit $h_\sigma = \lim_{s \to \infty} h_{\sigma^\frown s}$. Otherwise, σ is labeled by $\uparrow$. Then define h_T as the label of the root of T if it is defined. Observe that if h_σ is defined, then $h_\sigma \in B_{\operatorname{rank}_T(\sigma)}$. Indeed, one can compute a $B_{\operatorname{rank}_T(\sigma)}$-code of h_σ.

Definition 5.3. We call a pair (T, h) of a computable full-splitting well-founded tree T and a computable assignment h a *blueprint*.

One can easily see that $f \in B_\xi$ if and only if there exists a blueprint (T, h) such that $f = h_T$ and $\operatorname{rank}(T) = \xi$. We are now ready to prove the following:

Proposition 5.4. *For any computable ordinal ξ, we have $\mathsf{C}_{\mathsf{cof}}^{\emptyset^{(\xi)}} \equiv_T \emptyset^{(\xi+1)}$.*

Proof. It suffices to show that $\emptyset^{(\xi+1)} \leq_T \mathsf{C}_{\mathsf{cof}}^{\emptyset^{(\xi)}}$. As mentioned above, $\emptyset^{(\xi+1)}$ corresponds to $B_{1+\xi}$. Therefore, it suffices to show that $f \leq_T \mathsf{C}_{\mathsf{cof}}^{\emptyset^{(\xi)}}$ for any $f \in B_{1+\xi}$. Let (T, h) be a blueprint defining f as above. Since $h_T = f$ is defined, h_σ is also defined for any $\sigma \in T$ by definition.

We describe `Arthur`'s strategy for $f \leq_T \mathsf{C}_{\mathsf{cof}}^{\emptyset^{(\xi)}}$. Assume that `Merlin`'s first move is n, and the second and subsequent moves are $\sigma = \langle x_1, x_2, \ldots, x_\ell \rangle$. If σ is a leaf of T, then `Arthur` declares termination with $h_\sigma(n)$. Here, `Arthur` can use the information of h_σ since h_σ is computable. Assume that σ is not a leaf of T. Since the rank of T is $1 + \xi$, the rank of σ is at most $1 + \xi$ and thus the rank of any immediate successor $\sigma^\frown s$ of σ is less than $1 + \xi$. By the property of a blueprint, given $s \in \mathbb{N}$, one can compute a B_ζ-code of $h_{\sigma^\frown s} \in B_\zeta$ for some $\zeta < 1 + \xi$. In particular, $(s, n) \mapsto h_{\sigma^\frown s}(n)$ is $\emptyset^{(\xi)}$-computable.

Then consider the following program e: Given an input s, the computation searches for $t > s$ such that $h_{\sigma^\frown t}(n) \neq h_{\sigma^\frown s}(n)$. If such a t is found, the computation $\varphi_e^{\emptyset^{(\xi)}}(s)$

halts. Otherwise, $\varphi_e^{\emptyset^{(\xi)}}(s)$ never halts. Since h_σ is defined, $\lim_{s\to\infty} h_{\sigma^\frown s}(n)$ exists; that is, there exists s such that $h_{\sigma^\frown t}(n) = h_{\sigma^\frown s}(n)$ for any $t > s$. Hence, there are at most finitely many s such that $\varphi_e^{\emptyset^{(\xi)}}(s)$ halts. This means that $e \in \mathrm{dom}(\mathtt{C}_{\mathtt{cof}}^{\emptyset^{(\xi)}})$. Then, $\mathtt{Arthur}$ declares that the game is to continue, and uses e as the next query, that is, $\langle 0, e \rangle$ is $\mathtt{Arthur}$'s next move.

We claim that this is $\mathtt{Arthur}$'s winning strategy. To see this, if $\mathtt{Merlin}$'s second and subsequent moves are $\sigma = \langle x_1, x_2, \ldots, x_\ell \rangle$, then we inductively show that $h_T(n) = h_\sigma(n)$. We inductively assume that $h_T(n) = h_\sigma(n)$. As the next move, if $\mathtt{Arthur}$ declares that the game is to continue, and uses e as the next query, $\mathtt{Merlin}$ responds to this with some $t_{\ell+1} \in \mathtt{C}_{\mathtt{cof}}^{\emptyset^{(\xi)}}(e)$. This means that $h_{\sigma^\frown t_{\ell+1}}(n) = \lim_{s\to\infty} h_{\sigma^\frown s}(n) = h_\sigma(n)$. Hence, by the induction hypothesis, we have $h_{\sigma^\frown t_{\ell+1}}(n) = h_T(n)$.

If the history of $\mathtt{Merlin}$'s moves $\rho = \langle x_1, x_2, \ldots, x_k \rangle$ reaches a leaf of T, then $\mathtt{Arthur}$ declares termination with $h_\rho(n)$, and by the above property, we obtain $h_\rho(n) = h_T(n) = f(n)$. Hence, the procedure described above is shown to be $\mathtt{Arthur}$'s winning strategy. Consequently, we get $f \leq_T \mathtt{C}_{\mathtt{cof}}^{\emptyset^{(\xi)}}$. $\qquad\square$

As $\mathtt{C}_{\mathtt{cof}}^{\alpha} \leq_{LT} \mathtt{Cofinite}$ for any oracle $\alpha \in 2^{\mathbb{N}}$, this explains the reason why $\mathtt{Cofinite}$ is so powerful as an oracle. Interestingly, by the result in van Oosten [46, Theorem 2.2] mentioned above, one can observe that even if relativized by a tremendously powerful oracle α, $\mathtt{C}_{\mathtt{cof}}^{\alpha}$ never be able to compute a non-hyperarithmetic function; that is, for any non-hyperarithmetic f, we have $f \not\leq_T \mathtt{C}_{\mathtt{cof}}^{\alpha}$ no matter what an oracle α is. Roughly speaking, this is because a computational process beyond hyperarithmetic is not a finite approximation process, but an approximation process along an ordinal, which prevents us from using "time trick"; see e.g. [3]. As another remark, since $\mathtt{Cofinite}$ corresponds to giving an upper bound on an unknown value, it seems that van Oosten's characterization [46, Theorem 2.2] is somehow related to Solovay's result [44] that the computably encodable sets are exactly the hyperarithmetical ones.

5.3. **Asymptotic density.** As a candidate for another basic (non-modest) bilayer function, one that uses asymptotic density may come to mind. For a set $A \subseteq \mathbb{N}$, the *lower asymptotic density* of A is defined by

$$\underline{d}(A) = \liminf_{n\to\infty} \frac{|A \cap \{0, 1, \ldots, n-1\}|}{n}.$$

Then, for any real $\varepsilon \in [0, 1]$, we define the basic bilayer function $\mathtt{DenError}_\varepsilon$ as follows:

$$\mathrm{dom}(\mathtt{DenError}_\varepsilon) = \{(* \mid A) : A \subseteq \mathbb{N} \text{ and } \underline{d}(A) \geq 1 - \varepsilon\}$$

$$\mathtt{DenError}_\varepsilon(* \mid A) = A.$$

Obviously, we have $\mathtt{Cofinite} \leq_{LT}^1 \mathtt{DenError}_\varepsilon$ for any real $\varepsilon \in [0, 1]$ since the asymptotic density of a cofinite set is 1. The major difference between $\mathtt{Cofinite}$ and $\mathtt{DenError}_\varepsilon$ is the following property.

Observation 5.5. $\mathtt{Error}_{1/\ell} \leq_{LT}^1 \mathtt{DenError}_{1/\ell}$ *for any* $\ell \in \mathbb{N}$.

Proof. We define an outer reduction K as follows: For $n \in \mathbb{N}$ and $m < \ell$, put $K(n\ell + m) = m$. Then, for an input $(* \mid \{j\})$ for $\mathtt{Error}_{1/\ell}$, a secret inner reduction L is defined by $L(\{j\}) = \{n\ell + m : n \in \mathbb{N} \text{ and } j \neq m < \ell\}$. Note that the asymptotic density of

$L(\{j\})$ is $1 - 1/\ell$. Clearly, $y \in \mathtt{DenError}_{1/\ell}(L(\{j\})) = L(\{j\})$ implies $K(y) \neq j$; hence $K(y) \in \mathtt{Error}_{1/\ell}(* \mid \{j\})$. $\square$

We say that a partial multifunction $\mathtt{P} \colon\subseteq \mathbb{N} \rightrightarrows \mathbb{N}$ is hyperarithmetic if there exists a partial Π^1_1 function $f \colon\subseteq \mathbb{N} \to \mathbb{N}$ such that $f(n) \in \mathtt{P}(n)$ for any $n \in \mathrm{dom}(\mathtt{P})$.

Proposition 5.6. *Let $\mathtt{P}$ be a partial multifunction whose codomain is $\ell \in \mathbb{N}$ with $\ell > 0$. For any $\varepsilon < 1/(\ell+1)$, if $\mathtt{P} \leq_{LT} \mathtt{DenError}_\varepsilon$, then $\mathtt{P}$ is hyperarithmetic.*

To prove this, we need to show an auxiliary lemma. For a tree $T \subseteq \mathbb{N}^{<\mathbb{N}}$, let $\mathrm{succ}_T(t)$ be the set of all immediate successors of t in T. For a function $b \colon \mathbb{N}^{<\mathbb{N}} \to \mathbb{N}$, a tree T is b-*fat* if, for any $t \in T$ which is not a leaf, t has at least $b(t)$ immediate successors, i.e., $|\mathrm{succ}_T(t)| \geq b(t)$. For a function $b \colon \mathbb{N}^{<\mathbb{N}} \to \mathbb{N}$ and $\ell \in \mathbb{N}$, consider the function $\ell \cdot b \colon \sigma \mapsto \ell \cdot b(\sigma)$. The following is an analogue of Cenzer-Hinman [8, Proposition 2.9].

Lemma 5.7. *Let $b \colon \mathbb{N}^{<\mathbb{N}} \to \mathbb{N}$ be a function, T be an $(\ell \cdot b)$-fat finite tree, and L_T be the set of all leaves of T. Then, for any function $f \colon L_T \to \ell$ there exists a b-fat tree $S \subseteq T$ such that f is constant on the leaves of S.*

Proof. Since T is finite, one can assume that any $\sigma \in L_T$ has the same length. We prove the assertion by induction on the height k of T. Let $f \colon L_T \to \ell$ be given. If $k = 1$, then T has $\ell \cdot b(\varepsilon)$ leaves, where ε is the empty strings. As f is ℓ-valued, there are at least $b(\varepsilon)$ leaves on which f is constant.

Next, assume that $k > 1$. Then, for any $t \in T$ of length $k - 1$, define $g(t)$ as the least $j < \ell$ such that $|\{n \in \mathrm{succ}_T(t) : f(t^\frown n) = j\})| \geq b(t)$. Since T is $(\ell \cdot b)$-fat, we have $|\{n : f(t^\frown n) < \ell\}| \geq \ell \cdot b(t)$, so such a j exists. Note that g is a function from $T \cap \mathbb{N}^{k-1}$ to ℓ. By the induction hypotheis, there exists a b-fat tree $S^- \subseteq T \cap \mathbb{N}^{<k}$ such that g is constant on the leaves of S^-. Let $j < \ell$ be the unique value of g on the leaves L_S^- of S^-. Note that $L_S^- \subseteq T$. Then, define

$$S = S^- \cup \{t^\frown n : t \in L_S^- \ \wedge \ n \in \mathrm{succ}_T(t) \ \wedge \ f(t^\frown n) = j\}$$

By definition, clearly f is constant on the leaves of S. Moreover, for any $t \in L_S^-$, by our definition of g and S^-, we have

$$|\mathrm{succ}_S(t)| = |\{n \in \mathrm{succ}_T(t) : f(t^\frown n) = j\}| \geq b(t).$$

Therefore, S is b-fat. This concludes the proof. $\square$

Proof of Proposition 5.6. Let $(\tau \mid \eta)$ be a winning $\mathtt{Arthur}$-$\mathtt{Nimue}$ strategy witnessing $\mathtt{P} \leq_{LT} \mathtt{DenError}_\varepsilon$. Except for the first move n, $\mathtt{Merlin}$'s move is always a number $j \in \mathbb{N}$, which yields the tree $\mathbb{N}^{<\mathbb{N}}$ of all possible moves by $\mathtt{Merlin}$. Fix n, and then $\mathtt{Arthur}$'s strategy τ yields a partial computable function $\Phi^n_\tau \colon\subseteq \mathbb{N}^{<\mathbb{N}} \to \mathbb{N}$, where $\Phi^n_\tau(\sigma) \downarrow = u$ if and only if, after reading $\mathtt{Merlin}$'s moves σ, $\mathtt{Arthur}$'s strategy τ declares termination with u. $\mathtt{Nimue}$'s strategy η restricts $\mathtt{Merlin}$'s possible moves to a well-founded subtree T^n_η of $\mathbb{N}^{<\mathbb{N}}$ such that, for each $\sigma \in T^n_\eta$, if σ is a leaf then $\Phi^n_\tau(\sigma)$ is defined, and the lower asymptotic density of the set $A^n_\eta(\sigma) := \mathrm{succ}_{T^n_\eta}(\sigma)$ is at least $1 - \varepsilon$ since $\mathtt{Nimue}$ obeys the rule as long as $\mathtt{Merlin}$ obeys the rule, and this value is greater than $1 - 1/(\ell+1)$ since $\varepsilon < 1/(\ell+1)$. By the definition of lower asymptotic density, there exists $b(\sigma)$ such that,

for any $m \geq b(\sigma)$, we have $|A_\eta^n(\sigma) \cap m|/m > 1 - 1/(\ell+1)$. In particular, we have

$$|A_\eta^n(\sigma) \cap (\ell+1) \cdot b(\sigma)| > (\ell+1) \cdot b(\sigma) \cdot \left(1 - \frac{1}{\ell+1}\right) = \ell \cdot b(\sigma).$$

Fix such $b \colon \mathbb{N}^{<\mathbb{N}} \to \mathbb{N}$, where if σ is either a leaf of T_η^n or $\sigma \notin T_\eta^n$, then $b(\sigma)$ is arbitrary. Then, consider the tree T^b of $((\ell+1) \cdot b)$-bounded strings:

$$T^b = \{t \in \mathbb{N}^{<\mathbb{N}} : (\forall s < |t|)\ t(s) < (\ell+1) \cdot b(t \upharpoonright s)\}.$$

Since T^b is finite branching and T_η^n is well-founded, $T_\eta^{n,b} := T^b \cap T_\eta^n$ is finite by König's lemma. Moreover, Φ_τ^n is total on the leaves of $T_\eta^{n,b}$. Note that if σ is not a leaf of $T_\eta^{n,b}$ then $\mathrm{succ}_{T_\eta^{n,b}}(\sigma) = A_\eta^n(\sigma) \cap (\ell+1) \cdot b(\sigma)$. Hence, $T_\eta^{n,b}$ is $(\ell \cdot b)$-fat. Since Φ_τ^n is ℓ-valued, by Lemma 5.7, there exists a b-fat subtree S of $T_\eta^{n,b}$ of the same height such that Φ_τ^n is constant on the leaves of S. Hereafter, we write b as b_n since such a b satisfying the above density condition is depend on Merlin's first move n.

For a function $g \colon \mathbb{N}^{<\mathbb{N}} \to \mathbb{N}$, our algorithm $\Psi^g(n)$ searches for a g-fat finite tree $S \subseteq T^g$ such that Φ_τ^n is constant on the leaves of S, and returns the unique value j of Φ_τ^n on the leaves of S if such an S exists. Note that if $g(\sigma)$ is a correct witness for the above density condition for $A_\eta^n(\sigma)$, then such an S always exists. Now, we define Q as follows

$$Q(n,j) \iff (\forall f \colon \mathbb{N}^{<\mathbb{N}} \to \mathbb{N})(\exists g \geq f)\ [\Psi^g(n) \downarrow = j],$$

where by $f \leq g$ we mean that $f(\sigma) \leq g(\sigma)$ for any $\sigma \in \mathbb{N}^{<\mathbb{N}}$. Note that $\Psi^g(n)$ is defined only if $\Psi^g(n)$ succeeds in finding S, which means that the algorithm Ψ only reads g up to the height of S. Thus, the above predicate Q is Π_1^1.

We claim that $Q(n,j)$ implies $j \in \mathsf{P}(n)$. To see this, put $f(\sigma) = b_n(\sigma)$. If $g \geq f$ then we have $|A_\eta^n(\sigma) \cap (\ell+1) \cdot g(\sigma)| > \ell \cdot g(\sigma)$ by our choice of b_n. If $\Psi^g(n) \downarrow = j$ then the algorithm Ψ succeeds in finding a g-fat finite tree $S \subseteq T^g$ such that $\Phi_\tau^n(\rho) = j$ for any leaf ρ of S. Note that $\mathrm{succ}_S(\sigma) \subseteq (\ell+1) \cdot g(\sigma)$ by the definition of T^g, and $|\mathrm{succ}_S(\sigma)| \geq g(\sigma)$ since S is g-fat. This implies that $\mathrm{succ}_S(\sigma) \cap A_\eta^n(\sigma)$ is nonempty. Hence, $S \cap T_\eta^n$ has a common leaf ρ. Since $(\tau \mid \eta)$ is winning, and ρ is a leaf of T_η^n, we must have $\Phi_\tau^n(\rho) \in \mathsf{P}(n)$. By our choice of S, Φ_τ^n is constant on the leaves of S, and thus $\Psi^g(n)$ must be equal to $\Phi_\tau^n(\rho)$. This concludes $\Psi^g(n) = j \in \mathsf{P}(n)$.

We next claim that for any n there exists $j < \ell$ such that $Q(n,j)$. Otherwise, for any $j < \ell$ there exists f_j such that either $\Psi^g(n)$ is undefined or $\Psi^g(n) \neq j$ for any $g \geq f_j$. Then, put $h(\sigma) = \max\{b_n(\sigma), f_j(\sigma) : j < \ell\}$. As $h(\sigma) \geq b_n(\sigma)$, clearly, $h(\sigma)$ is a correct witness for the above density condition for $A_\eta^n(\sigma)$. Then, by the argument using Lemma 5.7 described above, the algorithm Ψ succeeds in finding S, so that $\Psi^h(n) \downarrow = j$ for some $j < \ell$. However, as $h \geq f_j$, this contradicts our assumption on f_j. Therefore, Q determines a total relation. Since Q is Π_1^1, by Δ_1^1-selection (see Moschovakis [31, 4B.5] or Sacks [40, Theorem II.2.3]), there exists a hyperarithmetic function $p \colon \mathbb{N} \to \mathbb{N}$ such that $Q(n, p(n))$ holds. By the first claim, this implies that $p(n) \in \mathsf{P}(n)$. □

In particular, $\mathtt{DenError}_\varepsilon$ and $\mathtt{Cofinite}$ have partly the same properties in the following sense.

Corollary 5.8. *Assume $\varepsilon < 1/2$. Then, for a function $f \colon \mathbb{N} \to \mathbb{N}$, $f \leq_{LT} \mathtt{DenError}_\varepsilon$ if and only if f is hyperarithmetic.*

Proposition 5.6 also shows that $\mathtt{LLPO}_{1/\ell}^{\alpha} \not\leq_{LT} \mathtt{DenError}_{1/(\ell+2)}$ for a sufficiently powerful oracle α. By Observation 4.1, this implies that $\mathtt{Error}_{1/\ell} \not\leq_{LT} \mathtt{DenError}_{1/(\ell+2)}$. Now it is natural to ask whether $\mathtt{Error}_{1/\ell} \leq_{LT} \mathtt{DenError}_{1/(\ell+1)}$ or not. One can answer this question by introducing the concept of hyperarithmetical reducibility for bilayer functions. First consider the one-query version.

Definition 5.9. Let f and g be bilayer functions. We say that f *is a one-query hyperarithmetically LT-reducible to* g (written $f \leq_{hLT}^{1} g$) if there exist partial Π_1^1 functions H and K and a partial function L such that for any $(n \mid c)$ and m,

$$m \in g(H(n) \mid L(n,c)) \implies K(n,m) \in f(n \mid c).$$

Let $\psi_e \colon \subseteq \mathbb{N} \to \mathbb{N}$ be the eth patrial Π_1^1 function (given by the canonical enumeration of all Π_1^1 sets). Then, consider the following partial multifunction:

$$\mathrm{dom}(\Pi_1^1\text{-}\mathtt{LLPO}_{m/k}) = \{e \in \mathbb{N} : |\{j < k : \psi_e(j) \downarrow\}| \leq m\},$$
$$\Pi_1^1\text{-}\mathtt{LLPO}_{m/k}(e) = \{0, \ldots, k-1\} \setminus \{j < k : \psi_e(j) \downarrow\}.$$

It is well-known that Π_1^1 is higher analogue of computable enumerability, i.e., a set is Π_1^1 if and only if there exists a hyperarithmetical enumeration procedure along a computable ordinal; see e.g. [40, 31]. The following is a hyperarithmetical analogue of Proposition 4.2:

Proposition 5.10. $\Pi_1^1\text{-}\mathtt{LLPO}_{m/k} \leq_{hLT}^{1} \mathtt{Error}_{m/k+1}$.

Proof. We define a secret inner reduction L as follows: For any $e \in \mathrm{dom}(\Pi_1^1\text{-}\mathtt{LLPO}_{m/k})$,

$$L(e) = \begin{cases} \{j < k \colon \psi_e(j) \downarrow\} & \text{if } |\{j < k \colon \psi_e(j) \downarrow\}| = m \\ \{j < k \colon \psi_e(j) \downarrow\} \cup \{k\} & \text{if } |\{j < k \colon \psi_e(j) \downarrow\}| < m \end{cases}$$

One can easily check that $(* \mid L(e))$ belongs to the domain of $\mathtt{Error}_{m/k+1}$. For an outer reduction K, define $K(e,j) = j$ for any $j < k$. To compute $K(e,k)$ along computable ordinal steps, wait for finding m many $j < k$ such that $\psi_e(j) \downarrow$. If it is found at some ordinal stage, then $K(e,k)$ is defined as the least $\ell < k$ such that $\ell \neq j$ for any such $j < k$. Otherwise, the computation never terminates, i.e., $K(e,k) \uparrow$. One can easily see that K is Π_1^1.

For readers who are not familiar with ordinal computability, we describe the details. Let $\mathcal{O} \subseteq \mathbb{N}$ be Kleene's system of ordinal notations; see e.g. [40, 10]. As Kleene's $\mathcal{O}$ is Π_1^1-complete, and the graph G_e of ψ_e is Π_1^1, there exists a many-one reduction p witnessing $G_e \leq_m \mathcal{O}$, where $\leq_m$ denotes many-one reducibility. Then, for $a \in \mathcal{O}$, one can consider the stage a approximation $\psi_e[a]$ of ψ_a; that is, $\psi_e(n)[a] \downarrow = m$ if and only if $p(n,m) <_{\mathcal{O}} a$. Note that $<_{\mathcal{O}}$ is c.e. (see e.g. Sacks [40, Theorem I.3.5]); hence $\psi_e[a] \downarrow$ is also a c.e. property. Then define G_K as follows:

$$(e,k,\ell) \in G_K \iff (\exists a \in \mathcal{O})\big[(\exists^{\geq m} j < k)\, \psi_e(j)[a] \downarrow$$
$$\wedge \ (\forall j < k)\, [\psi_e(j)[a] \downarrow \to \ j \neq \ell < k]\big]$$

One can easily see that G_K is Π_1^1 since $\mathcal{O}$ is Π_1^1. Hence, by Π_1^1-uniformization (see Sacks [40, Theorem II.2.3] or Moschovakis [31, 4B.4]), there exists a partial Π_1^1 function $K \colon \subseteq \mathbb{N}^2 \to \mathbb{N}$ such that $G_K(e,k,K(e,k))$ holds whenever $(e,k,\ell) \in G_K$ for some ℓ.

As in the proof of Proposition 4.2, one can see that $\langle L, K \rangle$ witnesses $\Pi_1^1\text{-LLPO}_{m/k} \leq_{hLT}^1$ $\text{Error}_{m/k+1}$. $\square$

Now we introduce the notion of hyperarithmetical reducibility for bilayer functions. Arthur's hyperarithmetic strategy is a code τ for a partial Π_1^1 function $h_\tau \colon \mathbb{N}^{<\mathbb{N}} \to \mathbb{N}$.

Definition 5.11. Let f and g be bilayer functions. We say that f *is hyperarithmetically LT-reducible to g* (written $f \leq_{hLT} g$) if there exists a hyperarithmetical winning Arthur-Nimue strategy for $\mathfrak{G}(f, g)$.

The following is an analogue of Proposition 5.6:

Proposition 5.12. *Let* P *be a partial multifunction whose codomain is* $\ell \in \mathbb{N}$ *with* $\ell > 0$. *For any* $\varepsilon < 1/(\ell + 1)$*, if* P $\leq_{hLT}$ $\text{DenError}_\varepsilon$*, then* P *is hyperarithmetic.*

Proof. The argument is the same as Proposition 5.6. Only the complexity of Q needs to be considered. If we consider a hyperarithmetical strategy, Φ_τ^n is no longer a computable function, but a Π_1^1 function. For this reason, Ψ is also Π_1^1. To see this, note that $\Psi^g(n)$ is defined to be j if and only if there exists a g-fat finite tree $S \subseteq T^g$ such that Φ_τ^n is defined and constant on the leaves of S and its unique value is j. This condition is Π_1^1 since S is finite and Φ_τ^n is Π_1^1. Moreover, as mentioned in the proof of Proposition 5.6, if $\Psi^g(n)$ is defined then the algorithm Ψ only reads g up to the height of S. Thus, $Q(n, j)$ holds if and only if for any f, there exists a finite string σ such that σ majorizes f up to $|\sigma|$ and $\Psi^\sigma(n) \downarrow = j$. This condition is Π_1^1. The rest follows the same argument as in Proposition 5.6. $\square$

Corollary 5.13. *For any* $\ell \geq 2$*,* $\text{Error}_{1/\ell} \leq_{LT} \text{DenError}_\varepsilon$ *if and only if* $1/\ell \leq \varepsilon$.

Proof. The backward direction follows from Observation 5.5. For the forward direction, assume that $1/\ell > \varepsilon$. It is clear that $\Pi_1^1\text{-LLPO}_{1/(\ell-1)}$ is not hyperarithmetic. In particular, by Proposition 5.12, we have $\Pi_1^1\text{-LLPO}_{1/(\ell-1)} \not\leq_{hLT} \text{DenError}_\varepsilon$ since $\varepsilon < 1/\ell$. However, by Proposition 5.10, we have $\Pi_1^1\text{-LLPO}_{1/(\ell-1)} \leq_{hLT} \text{Error}_{1/\ell}$. Hence, $\text{Error}_{1/\ell} \not\leq_{hLT}$ $\text{DenError}_\varepsilon$. $\square$

Note that the above proof also shows that $\Pi_1^1\text{-LLPO}_{1/\ell} \leq_{hLT} \text{DenError}_\varepsilon$ if and only if $1/(\ell + 1) \leq \varepsilon$.

The above results say nothing about DenError_0. Note that since the asymptotic density of a cofinite set is 1, we have $\text{Cofinite} \leq_{LT}^1 \text{DenError}_0$. We show that computability with density error 0 is strictly stronger than computability with finitely many error in the following sense:

Theorem 5.14. $\text{Cofinite} <_{LT} \text{DenError}_0$.

Proof. Suppose not. Then, there exists a winning Arthur-Nimue strategy $(\tau \mid \eta)$ witnessing $\text{DenError}_0 \leq_{LT} \text{Cofinite}$. Except for the first move $(* \mid A)$, Merlin's move is always a number $j \in \mathbb{N}$, which yields the tree $\mathbb{N}^{<\mathbb{N}}$ of all possible moves by Merlin. Here A is a secret input, which is invisible to Arthur. Hence, Arthur's strategy τ yields a partial computable function $\Phi_\tau \colon\subseteq \mathbb{N}^{<\mathbb{N}} \to \mathbb{N}$, where $\Phi_\tau(\sigma) \downarrow = u$ if and only if, after reading Merlin's moves σ, Arthur's strategy τ declares termination with u. For $h \colon \mathbb{N}^{<\mathbb{N}} \to \mathbb{N}$, consider the tree $\mathbb{N}^{<\mathbb{N}}[\geq h] = \{\sigma \in \mathbb{N}^{<\mathbb{N}} : (\forall n < |\sigma|)\ h(\sigma \restriction n) \leq \sigma(n)\}$. Nimue's strategy η restricts Merlin's possible moves to the tree $\mathbb{N}^{<\mathbb{N}}[\geq \eta_A]$, where $\eta_A(\sigma) = \eta(A^\frown \sigma)$, and

moreover, as τ is winning, the computation Φ_τ determines covers the tree $\mathbb{N}^{<\mathbb{N}}[\geq \eta_A]$; that is, for any infinite path x through $\mathbb{N}^{<\mathbb{N}}[\geq \eta_A]$ there exists an initial segment ξ of x such that $\Phi_\tau(\xi)$ is defined.

This ensures the existence of a function $h \colon \mathbb{N}^{<\mathbb{N}} \to \mathbb{N}$ such that the computation Φ_τ covers the tree $\mathbb{N}^{<\mathbb{N}}[\geq h]$. Consider the set B of all minimal strings $\xi \in \mathbb{N}^{<\mathbb{N}}[\geq h]$ such that $\Phi_\tau(\xi)$ is defined. Note that any infinite path x through $\mathbb{N}^{<\mathbb{N}}[\geq h]$ has an initial segment in B. Then, B yields a well-founded subtree $T = \{\zeta \in \mathbb{N}^{<\mathbb{N}} : (\exists \xi \in B)\ \zeta \preceq \xi\}$ of $\mathbb{N}^{<\mathbb{N}}[\geq h]$ so that B is the set of all leaves of T. One can see that if $\sigma \in T$ is not a leaf then the set $\mathrm{succ}_T(\sigma)$ of its immediate successors is cofinite. This is because, for any $n \geq h(\sigma)$, any infinite path extends $\sigma^\frown n \in \mathbb{N}^{<\mathbb{N}}[\geq h]$ has an initial segment $\rho \in B$. Then ρ is a leaf of T, and since $\sigma \in T$ is not a leaf, ρ must extend $\sigma^\frown n$. Hence we have $\sigma^\frown n \in T$ since a tree is $\preceq$-downward closed.

We label each node of this well-founded tree as follows: First, a leaf $\rho \in T$ is labeled by the value of $\Phi_\tau(\rho)$. If $\sigma \in T$ is not a leaf, then turn to its immediate successors. If σ has infinitely many immediate successors which have the same label c, then σ is also labeled by c. If there is no such label c, then σ is labeled by ∞.

Now, suppose that the label of the root of T is $c \neq \infty$. Then, `Merlin` plays $\mathbb{N} \setminus \{c\}$ as his first move, which has clearly asymptotic density 1. In the following, we assume that `Arthur` and `Nimue` follows their winning strategies τ and η, respectively. If `Nimue` reacts to the above move with z_0, search for $x_1 \geq z_0$ such that $\langle x_1 \rangle \in T$ and the label of $\langle x_1 \rangle$ is c. Such an x_1 exists, since for the former condition $\langle x_1 \rangle \in T$, recall that the set of immediate successors of a node in T is cofinite, and for the latter condition, the label of the root is c, so there are infinitely many immediate successors labeled by c. Then `Merlin` plays x_1 as his next move. Continuing this argument, `Merlin` can keep returning nodes of T with the same label c, and `Merlin`'s moves eventually reach a leaf of T. Reaching a leaf means that `Arthur` declares termination of the game with some value $\Phi_\tau(\rho)$, but since the label of this leaf ρ is c, the value $\Phi_\tau(\rho)$ must be c. Since $c \notin \mathtt{DenError}_0(\mathbb{N} \setminus \{c\})$ and `Merlin`'s first move is $\mathbb{N} \setminus \{c\}$, this means that `Merlin` wins the game, which contradicts our assumption that $(\tau \mid \eta)$ is a winning `Arthur`-`Nimue` strategy.

Thus, the root of T must be labeled by ∞. We say that a node $\sigma' \in T$ is a big sibling of a node $\sigma \in T$ if σ and σ' take the same value except for the last entry, and σ' is larger than σ for the last entry; that is, $\sigma(n) = \sigma'(n)$ for any $n < |\sigma| - 1$ and $\sigma(|\sigma| - 1) < \sigma'(|\sigma| - 1)$. We also say that a node $\sigma \in T$ is decisive if all proper initial segments of σ are labeled by ∞, but σ and all big siblings of σ are labeled by some values in $\mathbb{N}$. Note that the root of T is not decisive as it is labeled by ∞, so any decisive node has a proper initial segment. Let $(\alpha_s)_{s \in \mathbb{N}}$ be a list of all decisive nodes of T. First put $d_0 = 1$. At stage s, assume that d_s has already been defined. The immediate predecessor α_s^- of α_s is labeled by ∞ since α_s is decisive. The label ∞ of α_s^- means that, for any $c \in \mathbb{N}$, there are only finitely many siblings of α_s labeled by c. Therefore, by the pigeonhole principle, α_s has a big sibling labeled by some $c_s > d_s$. As α_s is decisive, c_s must be a finite value. Then, put $d_{s+1} = 2c_s$.

Now `Merlin` plays $\mathbb{N} \setminus \{c_s : s \in \mathbb{N}\}$ as his first move, which has asymptotic density 1 since $c_{s+1} > 2c_s$ for any s by our construction. At the round $n + 1$, assume that the history of `Merlin`'s previous moves is $x_1, \ldots, x_n$, and `Nimue`'s previous move is z_n. First consider the case that the label of $\sigma_n = \langle x_1, \ldots, x_n \rangle$ is ∞. If σ_n has infinitely many
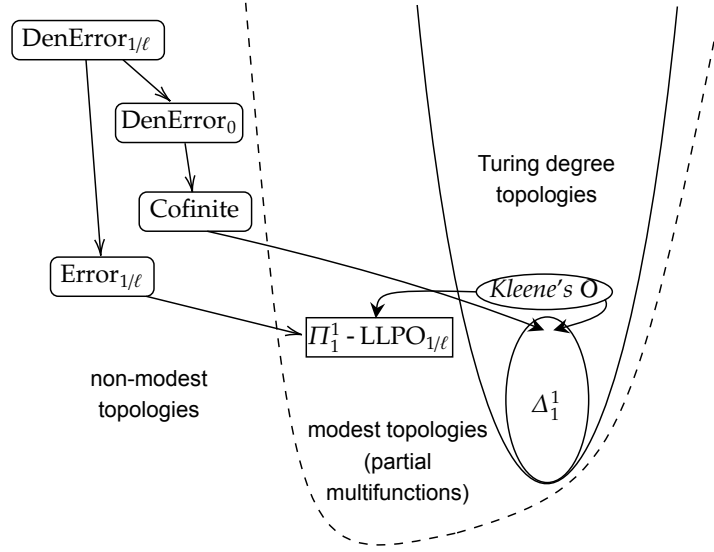
FIGURE 2. Higher parts of Lawvere-Tierney topologies on the effective topos

immediate successors labeled by ∞, then as his next move Merlin plays $x_{n+1} \geq z_n$ so that $\sigma_n{}^\frown x_{n+1} \in T$ is labeled by ∞. Otherwise, there are only finitely many immediate successors of σ_n labeled by ∞, and thus, there exists a decisive immediate successor of σ_n of the form $\sigma_n{}^\frown z$ for some $z \geq z_n$. Then we must have $\alpha_s = \sigma_n{}^\frown z$ for some s. As seen above, α_s has a big sibling $\alpha'_s = \sigma_n{}^\frown z'$ labeled by c_s. As his next move, Merlin plays the last entry $x_{n+1} := z'$ of α'_s. Note that the history $\alpha'_s = \langle x_1, \ldots, x_n, x_{n+1} \rangle$ of moves is now labeled by c_s. Next consider the case that the label of $\sigma_n = \langle x_1, \ldots, x_n \rangle$ has already become a finite value $c \in \mathbb{N}$. In this case, σ_n has infinitely many immediate successors labeled by c, and then as his next move Merlin can play $x_{n+1} \geq z_n$ so that $\sigma_n{}^\frown x_{n+1} \in T$ is labeled by c.

As this play follows a winning Arthur-Nimue strategy, Arthur declares termination at some round. Then, the history of Merlin's moves eventually reaches a leaf of T which is labeled by a finite value. Hence, the history of Merlin's moves is labeled by a finite value at some round, and once the label becomes a finite value, our construction of Merlin's strategy ensures that the value of the label does not change after that. Indeed, Merlin's strategy described above stabilizes the labels of the histories of Merlin's moves to c_s for some s. Therefore, the history of Merlin's moves eventually reaches a leaf of T which is labeled by c_s, which turns out to be (the second coordinate of) Arthur's last move since the leaf of T is labeled by the value of Φ_τ on it. However, $c_s \notin \text{DenError}_0(\mathbb{N} \setminus \{c_s : s \in \mathbb{N}\})$ and Merlin's first move is $\mathbb{N} \setminus \{c_s : s \in \mathbb{N}\}$. This means that Merlin wins the game, which contradicts our assumption that $(\tau \mid \eta)$ is a winning Arthur-Nimue strategy. Consequently, there exists no winning Arthur-Nimue strategy; hence $\text{Cofinite} <_{LT} \text{DenError}_0$. □

Hence, we get the strict hierarchy of computability with error density ε:

$$\text{Cofinite} <_{LT} \text{DenError}_0 <_{LT} \ldots <_{LT} \text{DenError}_{1/3} <_{LT} \text{DenError}_{1/2} \equiv_{LT} \neg\neg.$$

Figure 2 summarizes some of basic implications about the $\leq_{\mathrm{rea}}$-ordering on the Lawvere-Tierney topologies (around hyperarithmetic Turing topologies) on the effective topos, where $A \to B$ means $B \leq_{LT} A$ (or $B^{\supset \to} \leq_{\mathrm{rea}} A^{\supset \to}$).

6. Future work

One may come up with other basic bilayer functions not mentioned in this article, but we do not know which ones are non-trivial and interesting. It is a vague question, but finding interesting basic non-modest bilayer functions is a big problem in itself.

Question 1. *Is there any other interesting basic non-modest bilayer function?*

In Section 5.1, we have seen that the $\mathbb{N}$-version of weak weak König's lemma, WWKL, is Turing equivalent to LLPO. Due to this kind of phenomenon, unlike $\mathbb{N}^{\mathbb{N}}$-computation, it is difficult to find a nontrivial partial multifunction in the context of $\mathbb{N}$-computation. There are partial multifunctions on $\mathbb{N}$ not mentioned so far, such as *all-or-unique choice* $\mathtt{AoUC}_X$ on X (see e.g. [24]). However, the $\mathbb{N}$-version of $\mathtt{AoUC}_{2^{\mathbb{N}}}$ turns out to be Turing equivalent to LLPO by the same argument as in Section 5.1, and $\mathtt{AoUC}_{\mathbb{N}}$ is Turing equivalent to the halting problem by considering enumeration time functions as in Section 2.1.

Question 2. *Is there any other natural partial multifunction on $\mathbb{N}$ whose Turing degree strictly lies between the computable ones and the halting problem?*

In this article, we have focused on topologies on the effective topos; however "*the world of computable mathematics*" for modern computability theorists seems to be the Kleene-Vesley topos rather than the effective topos. As indicated in Section 2.2, the structure of Lawvere-Tierney topologies on the Kleene-Vesley topos seems isomorphic to the bilayer version of generalized Weihrauch reducibility. In the Kleene-Vesley context, modest bilayer functions have already been studied in depth. There seem several results on non-modest functions as well: In the study of (computability-theoretic) reverse mathematics on Ramsey's theorem, the notions of encodability [34, 9] and omniscience reducibility [29, 17, 15] have been introduced, and interestingly, these notions seem to play a role in transforming modest functions into non-modest functions. Although only one-query reducibility has been studied yet in such a context, there seems to be no doubt that non-modest functions are important in the study of reverse mathematics on Ramsey's theorem. This structure should also be explored in depth in the future.

Question 3. *Study the LT-degrees of basic non-modest bilayer functions on $\mathbb{N}^{\mathbb{N}}$.*

As another topos, the realizability topos $\mathbf{RT}(K_2)$ induced by Kleene's second algebra corresponds to "*the world of continuous mathematics*," and a Lawvere-Tierney topology on the topos is a kind of data that indicates how much discontinuity to add to the world. One can see that the structure of Lawvere-Tierney topologies on the topos $\mathbf{RT}(K_2)$ is isomorphic to the bilayer version of generalized continuous Weihrauch reducibility.

In general, there are many other toposes that are related to computability theory and (effective) descriptive set theory. As mentioned in Kihara [23], any Σ^*-pointclass (see [31]) yields a (relative) partial combinatory algebra, which induces a topos. If the pointclass $\underset{\sim}{\mathbf{\Pi}}^1_1$ is used as a seed, a topos corresponding to "the world of Borel measurable mathematics" will be created, and if the pair $(\Pi^1_1, \mathbf{\Pi}^1_1)$ is used, a topos corresponding to "the world of effective Borel measurable mathematics" will be created. These lead us to

the study of "Lawvere-Tierney topologies for (effective) descriptive set theorists." The topologies in this case are some sort of data that indicate how much non-Borel objects to add to the world.

It may also be reasonable to study these structures in the context of synthetic descriptive set theory [36]. We leave the exploration of these structures as a future task.

Question 4. *Investigate the structure of Lawvere-Tierney topologies on relative realizability toposes.*

For the specific toposes mentioned above (such as the Kleene-Vesley topos), their Lawvere-Tierney topologies can be characterized by *LT*-reducibility, using the proof of this article as is, but does the same result hold for an arbitrary relative realizability topos?

Question 5. *Does an analogue of Corollary 3.5 hold for any relative realizability topos?*

Finally, coincidentally, instance reducibility in Bauer [2] and the ordering on computably monotone functions (Section 3.2) have the exact same structure, but what is the connection between them? For instance, can we characterize the ordering on Lawvere-Tierney topologies by a sort of iterated instance reducibility?

Question 6. *Characterize the ordering on Lawvere-Tierney topologies on the effective topos (or other relative realizability toposes) using a variant of instance reducibility.*

References

[1] Paul-Elliot Anglès d'Auriac and Takayuki Kihara. A comparison of various analytic choice principles. to appear in Journal of Symbolic Logic, arXiv:1907.02769, 2019.

[2] Andrej Bauer. Instance reducibility and Weihrauch degrees. arXiv:2106.01734, 2021.

[3] Laurent Bienvenu, Noam Greenberg, and Benoit Monin. Bad oracles in higher computability and randomness. *Israel J. Math.*, 241(1):229–276, 2021.

[4] Vasco Brattka and Guido Gherardi. Effective choice and boundedness principles in computable analysis. *Bull. Symbolic Logic*, 17(1):73–117, 2011.

[5] Vasco Brattka, Guido Gherardi, and Rupert Hölzl. Probabilistic computability and choice. *Inform. and Comput.*, 242:249–286, 2015.

[6] Vasco Brattka, Guido Gherardi, and Arno Pauly. *Weihrauch Complexity in Computable Analysis*, pages 367–417. Springer International Publishing, Cham, 2021.

[7] Vasco Brattka and Arno Pauly. Computation with advice. In Xizhong Zheng and Ning Zhong, editors, Proceedings Seventh International Conference on *Computability and Complexity in Analysis,* Zhenjiang, China, 21-25th June 2010, volume 24 of *Electronic Proceedings in Theoretical Computer Science*, pages 41–55. Open Publishing Association, 2010.

[8] Douglas Cenzer and Peter G. Hinman. Degrees of difficulty of generalized r.e. separating classes. *Arch. Math. Logic*, 46(7-8):629–647, 2008.

[9] Peter Cholak and Ludovic Patey. Thin set theorems and cone avoidance. arXiv:1812.00188, 2018.

[10] Chi Tat Chong and Liang Yu. *Recursion theory*, volume 8 of *De Gruyter Series in Logic and its Applications*. De Gruyter, Berlin, 2015. Computational aspects of definability, With an interview with Gerald E. Sacks.

[11] S. Barry Cooper. *Computability Theory*. Chapman & Hall/CRC, Boca Raton, FL, 2004.

[12] Hannes Diener. Constructive reverse mathematics, 2018. Habilitationsschrift, University of Siegen.

[13] Hannes Diener and Hajime Ishihara. *Bishop-Style Constructive Reverse Mathematics*, pages 347–365. Springer International Publishing, Cham, 2021.

[14] Rodney G. Downey and Denis R. Hirschfeldt. *Algorithmic Randomness and Complexity*. Theory and Applications of Computability. Springer, New York, 2010.

[15] Damir Dzhafarov and Ludovic Patey. COH, SRT_2^2, and multiple functionals. arXiv: 1905.00321, 2019.

[16] Damir D. Dzhafarov. Strong reductions between combinatorial principles. *J. Symb. Log.*, 81(4):1405–1431, 2016.

[17] Damir D. Dzhafarov, Denis R. Hirschfeldt, and Sarah C. Reitzes. Reduction games, provability, and compactness. arXiv:2008.00907, 2020.

[18] Jun Le Goh. Compositions of multivalued functions. *Computability*, 9(3-4):231–247, 2020.

[19] Jun Le Goh, Arno Pauly, and Manlio Valenti. Finding descending sequences through ill-founded linear orders. to appear in Journal of Symbolic Logic, arXiv:2010.03840, 2021.

[20] Edward R. Griffor, editor. *Handbook of Computability Theory*, volume 140 of *Studies in Logic and the Foundations of Mathematics*. North-Holland Publishing Co., Amsterdam, 1999.

[21] Denis R. Hirschfeldt and Carl G. Jockusch, Jr. On notions of computability-theoretic reduction between Π_2^1 principles. *J. Math. Log.*, 16(1):1650002, 59, 2016.

[22] J. M. E. Hyland. The effective topos. In *The L.E.J. Brouwer Centenary Symposium (Noordwijkerhout, 1981)*, volume 110 of *Stud. Logic Foundations Math.*, pages 165–216. North-Holland, Amsterdam-New York, 1982.

[23] Takayuki Kihara. Degrees of incomputability, realizability and constructive reverse mathematics. arXiv:2002.10712, 2020.

[24] Takayuki Kihara and Arno Pauly. Dividing by zero—how bad is it, really? In *41st International Symposium on Mathematical Foundations of Computer Science*, volume 58 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 58, 14. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2016.

[25] Sori Lee. Subtoposes of the effective topos. arXiv:1112.5325, 2011.

[26] Sori Lee and Jaap van Oosten. Basic subtoposes of the effective topos. *Ann. Pure Appl. Logic*, 164(9):866–883, 2013.

[27] Vladimir Lifschitz. CT_0 is stronger than CT_0!. *Proc. Amer. Math. Soc.*, 73(1):101–106, 1979.

[28] Saunders Mac Lane and Ieke Moerdijk. *Sheaves in Geometry and Logic*. Universitext. Springer-Verlag, New York, 1994. A first introduction to topos theory, Corrected reprint of the 1992 edition.

[29] Benoit Monin and Ludovic Patey. Π_1^0-encodability and omniscient reductions. *Notre Dame J. Form. Log.*, 60(1):1–12, 2019.

[30] Antonio Montalbán. Martin's conjecture: a classification of the naturally occurring Turing degrees. *Notices Amer. Math. Soc.*, 66(8):1209–1215, 2019.

[31] Yiannis N. Moschovakis. *Descriptive Set Theory*, volume 155 of *Mathematical Surveys and Monographs*. American Mathematical Society, Providence, RI, second edition, 2009.

[32] Eike Neumann and Arno Pauly. A topological view on algebraic computation models. *J. Complexity*, 44:1–22, 2018.

[33] Piergiorgio Odifreddi. *Classical Recursion Theory*, volume 125 of *Studies in Logic and the Foundations of Mathematics*. North-Holland Publishing Co., Amsterdam, 1989. The theory of functions and sets of natural numbers, With a foreword by G. E. Sacks.

[34] Ludovic Patey. Ramsey-like theorems and moduli of computation. *The Journal of Symbolic Logic*, pages 1–26, 2020.

[35] Arno Pauly. Many-one reductions and the category of multivalued functions. *Math. Structures Comput. Sci.*, 27(3):376–404, 2017.

[36] Arno Pauly and Matthew de Brecht. Descriptive set theory in the category of represented spaces. In *2015 30th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2015)*, pages 438–449. IEEE Computer Soc., Los Alamitos, CA, 2015.

[37] Andrew M. Pitts. *The Theory of Triposes*. PhD thesis, The University of Cambridge, 1981.

[38] Fred Richman. Polynomials and linear transformations. *Linear Algebra Appl.*, 131:131–137, 1990.

[39] Hartley Rogers, Jr. *Theory of Recursive Functions and Effective Computability*. MIT Press, Cambridge, MA, second edition, 1987.

[40] Gerald E. Sacks. *Higher Recursion Theory*, volume Volume 2 of *Perspectives in Mathematical Logic*. Springer-Verlag, Berlin, 1990.
[41] Stephen G. Simpson. An extension of the recursively enumerable Turing degrees. *J. Lond. Math. Soc. (2)*, 75(2):287–297, 2007.
[42] Stephen G. Simpson. *Subsystems of Second Order Arithmetic*. Perspectives in Logic. Cambridge University Press, 2009.
[43] Robert I. Soare. *Recursively Enumerable Sets and Degrees*. Perspectives in Mathematical Logic. Springer-Verlag, Berlin, 1987. A study of computable functions and computably generated sets.
[44] Robert M. Solovay. Hyperarithmetically encodable sets. *Trans. Amer. Math. Soc.*, 239:99–122, 1978.
[45] Jaap van Oosten. *Realizability: an introduction to its categorical side*, volume 152 of *Studies in Logic and the Foundations of Mathematics*. Elsevier B. V., Amsterdam, 2008.
[46] Jaap van Oosten. Realizability with a local operator of A. M. Pitts. *Theoret. Comput. Sci.*, 546:237–243, 2014.
[47] Linda Westrick. A note on the diamond operator. *Computability*, 10(2):107–110, 2021.
[48] Martin Ziegler. Real computation with least discrete advice: a complexity theory of nonuniform computability with applications to effective linear algebra. *Ann. Pure Appl. Logic*, 163(8):1108–1139, 2012.

(Takayuki Kihara) GRADUATE SCHOOL OF INFORMATICS, NAGOYA UNIVERSITY, NAGOYA, 464-8601 JAPAN
E-mail address: kihara@i.nagoya-u.ac.jp