Asymptotic Logical Uncertainty and the Benford Test

In: ***Artificial General Intelligence: 9th International Conference, AGI 2016, New York, NY, USA, July 16–19, 2016. Proceedings,*** **9782:202–211.**
Lecture Notes in Artificial Intelligence. Springer International Publishing*

Scott Garrabrant[1,2], Tsvi Benson-Tilsen[1,3], Siddharth Bhaskar[2], Abram Demski[1,4], Joanna Garrabrant, George Koleszarik, and Evan Lloyd[2]

[1] Machine Intelligence Research Institute
[2] University of California, Los Angeles
[3] University of California, Berkeley
[4] University of Southern California

Abstract. Almost all formal theories of intelligence suffer from the problem of *logical omniscience*, the assumption that an agent already knows all consequences of its beliefs. *Logical uncertainty* codifies uncertainty about the consequences of existing beliefs. This implies a departure from beliefs governed by standard probability theory. Here, we study the asymptotic properties of beliefs on quickly computable sequences of logical sentences. Motivated by an example we call the Benford test, we provide an approach which identifies when such subsequences are indistinguishable from random, and learns their probabilities.

1 Introduction

Probabilistic reasoning about deterministic structures is a challenging case of uncertain reasoning which has received relatively little attention. This is the subject of *logical uncertainty* as defined in e.g. [1]: "any realistic agent is necessarily uncertain not only about its environment or about the future, but also about the logically necessary consequences of its beliefs." Being able to produce well-reasoned guesses about the results of programs before running them could provide valuable information for heuristic search, such as planning in complex environments or automatic programming. This kind of uncertainty can also be of wider interest. For example, [2] provides a call to arms for the development of numerical algorithms which provide information about the uncertainty in their results. [3] discusses the use of probabilistic information generated by machine learning to aid static program analysis for optimization.

* The final publication is available at Springer via http://dx.doi.org/10.1007/978-3-319-41649-6_20.

One of the purest examples of this type of reasoning in humans is the formation of mathematical conjectures. Mathematical intuitions behave in a way that is largely consistent with the standard probability axioms [4]. However, they cannot be entirely so: probability theory requires belief in anything which follows logically from your current beliefs, and so cannot represent uncertainty about what may later be proved. This is known as the problem of *logical omniscience*, and is the main challenge faced by a theory of logical uncertainty [5, 6, 7, 8].

Several proposals have addressed this problem by considering sequences of probability assignments[5] which converge to logically omniscient distributions, enforcing more and more constraints imposed by the probability axioms in the limit of unbounded computing resources [9, 1, 10]. With such a theoretical model in hand, one might think that the practical problem of logical uncertainty amounts to approaching this limit as quickly as possible. However, this is a fairly weak constraint on the behavior of beliefs at finite time.

If we're asked to quickly give a probability for a mathematical question, and the question belongs to a class of questions which have been true 25% of the time, it seems we should give a probability close to 0.25. This type of reasoning is ignored if our only aim is to converge to a good probability distribution overall. Each individual question will converge to probability one or zero. This seems to ignore an essential part of the problem. So, we take a different approach. We consider the limit of a sequence of *sentences* within a *single* assignment of probabilities, rather than the limit of probabilities within a sequence of assignments. We call this approach *asymptotic logical uncertainty*. Our goal is not to provide an algorithm which may be used in an AGI directly, but rather, to illustrate a new desirable (and achievable) property of uncertain reasoning for AGI.

Section 2 discusses further related work. In Section 3, we define the Benford test as a concrete example of the type of reasoning we wish to model. Section 4 defines *irreducible patterns*, a concept used to define a general case where this sort of reasoning is justified. Section 5 proposes a learning algorithm to solve the problem in the general case, and Section 6 proves that the method is successful. Section 7 concludes.

2 Related Work

The most often-cited work relating probability to logic is almost certainly that of Cox [11], which shows that under certain desirable assumptions, probability theory is the only possible generalization of Boolean algebra. Other early work concerning measures over Boolean algebras include [12, 13]. This has since been extended to first-order logic [14, 15], and from there to other settings [16, 17, 18]. However, most of this work does not address computability.

An early articulation of the problem of logical omniscience was [5]. Many approaches have attempted to deal with this through theories of inconsistent structures, including [7, 8]. The approach here is also related to online sequence

[5] We will use the term "probability" to refer to degrees of belief generally, whether or not the probability axioms are obeyed.

learning using expert advice, to predict a sequence of observations almost as well as a given set of advisors [19], especially experts on sub-sequences as in [20].

3 The Benford Test

Benford's law states that in naturally occurring numbers, the leading digit $d \in \{1, \ldots, 9\}$ of that number in base 10 occurs with probability $\log_{10}(1 + \frac{1}{d})$. Many mathematical sequences have been shown to have frequencies of first digits that satisfy Benford's law [21]. In particular, the frequencies of the first digits of powers of 3 provably satisfy Benford's law.

The function $3 \uparrow^n k$ is a fast-growing function defined by $3 \uparrow^1 k = 3^k$, $3 \uparrow^n 1 = 3$, and $3 \uparrow^n k = 3 \uparrow^{n-1} (3 \uparrow^n (k-1))$. $3 \uparrow^n k$ is very large, and first digit of $3 \uparrow^n k$ is probably very difficult to compute. It is unlikely that the first digit of $3 \uparrow^3 3$ will ever be known.

If asked to quickly assign a probability to the sentence "The first digit of $3 \uparrow^3 3$ is a 1," it seems the only reasonable answer would be to treat it as a power of three and reply $\log_{10}(2) \approx .30103$, as dictated by Benford's law. Note that the sentence is either true or false; there are no random variables. The probability here represents a reasonable guess in the absence of enough time or resources to compute $3 \uparrow^3 3$.

We define the Benford test to formalize this reasoning.[6] Throughout the paper, let the time-bound $T(N)$ be an increasing function in the range of $N \leq T(N) \leq 3 \uparrow^k N$ for some fixed k, and $R(N) = T(N)N^4 \log T(N)$ a larger time-bound.

Definition 1 *Let M be a Turing machine which on input N runs in time $O(R(N))$ and outputs a probability $M(N)$, which represents the probability assigned to ϕ_N. We say that M passes the* Benford test *if*

$$\lim_{n \to \infty} M(s_n) = \log_{10}(2)\,, \tag{1}$$

where ϕ_{s_n} = "The first digit of $3 \uparrow^n 3$ is a 1."

It is easy to pass the Benford test by hard-coding in the probability. It is more difficult to pass the Benford test in a natural way. That the best probability to assign to ϕ_{s_n} is $\log_{10}(2)$ depends not only on the fact that the frequency with which ϕ_{s_n} is true tends toward $\log_{10}(2)$, but also on the fact that the sequence of truth-values of ϕ_{s_n} contains no patterns that can be used to quickly compute a better probability on some subsequence. We therefore assume that this sequence of truth-values is indistinguishable from a sequence produced by a coin that outputs "true" with probability $\log_{10}(2)$. Formally, we are assuming that $S = \{s_n | n \in \mathbb{N}\}$ is an *irreducible pattern* with probability $\log_{10}(2)$, as defined in the next section.

[6] The test presumes that the frequencies of the first digits in the sequence $3 \uparrow^n 3$ satisfy Benford's law. Though this seems likely, the conjecture is not too important; any sufficiently fast-growing sequence satisfying Benford's law could serve as an example.

4 Irreducible Patterns

Let $\phi_1, \phi_2, \ldots$ be a simple enumeration of all sentences in first order logic over ZFC. Fix a universal Turing machine U and an encoding scheme for machines, and let $U(M, x)$ denote running the machine U to simulate M with input x.

Definition 2 [7] *Let $S \subseteq \mathbb{N}$ be an infinite subset of natural numbers such that ϕ_N is provable or disprovable in ZFC for all $N \in S$, and there exists a Turing machine Z such that $U(Z, N)$ runs in time $T(N)$ and accepts N if and only if $N \in S$.*

We say that S is an irreducible pattern with probability p *if there exists a constant c such that for every positive integer $m \geq 3$ and every Turing machine W expressible in $K(W)$ bits, if*

$$S' = \{N \in S \mid U(W, N) \text{ accepts in time } T(N)\} \tag{2}$$

has at least m elements and $r(m, W)$ is the probability that ϕ_N is provable when N is chosen uniformly at random from the first m elements of S', we have

$$|r(m, W) - p| < \frac{cK(W)\sqrt{\log\log m}}{\sqrt{m}}\,. \tag{3}$$

The intuition behind the formula is that the observed frequency $r(m, W)$ for any sequence S' we select should not stray far from p. The right hand side of the inequality needs to shrink slowly enough that a true random process would stay within it with probability 1 (given choice of c sufficiently large to accommodate initial variation). The law of the iterated logarithm gives such a formula, which is also tight in the sense that we cannot replace it with a formula which diminishes more quickly as a function of m.

Proposition 1 *If provability in Definition 2 were decided randomly, such that for each $N \in S$ the sentence ϕ_N is independently called "provable" with probability p and "disprovable" otherwise, then S would almost surely be an irreducible pattern with probability p.*

Proof. Omitted due to space limitations.[8]

We now use the concept of irreducible patterns to generalize the Benford test.

[7] We tailored this definition of irreducible pattern to our needs. The theory of algorithmic randomness may offer alternatives. However, algorithmic randomness generally considers all computable tests and focuses on the case where $p = \frac{1}{2}$ [22, 23]. We believe that any reasonable definition inspired by algorithmic randomness would imply Definition 2.

[8] See pre-print version [24] for the full proof.

Definition 3 *Let M be a Turing machine which on input N runs in time $O(R(N))$ and outputs a probability $M(N)$, which represents the probability assigned to ϕ_N. We say that M passes the* generalized Benford test *if*

$$\lim_{\substack{N\to\infty \\ N\in S}} M(N) = p\,, \tag{4}$$

whenever S is an irreducible pattern with probability p.

Note that if we conjecture that the S from Definition 1 is an irreducible pattern with probability $\log_{10}(2)$, then any M which passes the generalized Benford test also passes the Benford test.

5 A Learning Algorithm

We now introduce an algorithm $A_{L,T}$ that passes the generalized Benford test (see Algorithm 1). The general idea behind the algorithm is to make a prediction for a sentence by searching for an irreducible pattern which it belongs to (represented by the program X). To be sure that a pattern is irreducible, we must also search for any subsequences (represented by Y) which have significantly different probabilities. In effect, we are trying to predict an event by finding a *reference class* which the event belongs to. A reference class which is simple and passes tests for pseudo-randomness is chosen, since this indicates that we are unlikely to do better by choosing a different reference class.

Let L be the Turing machine which accepts on input N if ZFC proves ϕ_N, rejects on input N if ZFC disproves ϕ_N, and otherwise does not halt. For convenience, in Algorithm 1, we define $\log q = 1$ for $q < 2$.
Let $TM(N)$ be the set of all Turing machines X expressible in at most $\log N$ bits such that $U(X, N)$ accepts in time at most $T(N)$. The encoding of Turing machines must be prefix-free, which in particular means that no Turing machine is encoded in 0 bits. Let J_N denote the set of rational numbers of the form $\frac{j}{N}$ with $j = 0, \ldots, N$.

For X and Y Turing machines, let $K(X)$ be the number of bits necessary to encode X. Let $S'(X, Y)$ be the subset of natural numbers i which are accepted by both $U(X, i)$ and $U(Y, i)$ in time at most $T(i)$. Let $Q_N(X, Y)$ be the greatest number less than or equal to N such that for every s in the first $Q_N(X, Y)$ elements of S', $U(L, s)$ halts in time $T(N)$. Let $F_N(X, Y)$ be the proportion of the first $Q_N(X, Y)$ elements of S' which L accepts. Let

$$B_N(X, Y, P) = \max\left(K(X), \frac{|F_N(X, Y) - P|\sqrt{Q_N(X, Y)}}{K(Y)\sqrt{\log\log Q_N(X, Y)}}\right). \tag{5}$$

Lemma 1 *The output of $A_{L,T}$ on input N is in*

$$\arg\min_{P\in J_N} \max_{Y\in TM(N)} \min_{X\in TM(N)} B_N(X, Y, P)\,. \tag{6}$$

Proof. Omitted due to space limitations. [9]

The code is not optimized for computational efficiency. The following proposition is just to ensure that the runtime is not far off from $T(N)$.

Proposition 2 *The runtime of* $A_{L,T}(N)$ *is in* $O(R(N)) = O(T(N)N^4 \log T(N)))$.

Proof. Simulating U on any input for T time steps can be done in time $cT \log T$ for some fixed constant c [25]. The bulk of the runtime comes from simulating Turing machines on lines 8, 13, 14, and 16. Each of these lines takes at most $cT(N) \log T(N)$ time, and we enter each of these lines at most N^4 times. Therefore, the program runs in time $O(T(N)N^4 \log T(N))$.

Algorithm 1 $A_{L,T}(N)$

```
 1: P = 0
 2: M = N
 3: for j = 0, ..., N do
 4:     M_Y = 0
 5:     for Y a Turing machine expressible in K_Y < log N bits do
 6:         M_X = N
 7:         for X a Turing machine expressible in K_X < log N bits do
 8:             if U(X, N) and U(Y, N) both accept in time T(N) then
 9:                 A = 0
10:                 R = 0
11:                 i = 1
12:                 while i ≤ N do
13:                     if U(X, i) and U(Y, i) both accept in time T(i) then
14:                         if U(L, i) accepts in time T(N) then
15:                             A = A + 1
16:                         else if U(L, i) rejects in time T(N) then
17:                             R = R + 1
18:                         else
19:                             i = N
20:                     i = i + 1
21:                 F = A/(A + R)
22:                 Q = A + R
23:                 if max(K_X, |F − j/N|√Q / (K_Y √(log log Q))) < M_X then
24:                     M_X = max(K_X, |F − j/N|√Q / (K_Y √(log log Q)))
25:         if M_X > M_Y then
26:             M_Y = M_X
27:     if M_Y < M then
28:         M = M_Y
29:         P = j/N
30: return P
```

[9] See pre-print version [24] for the full proof.

6 Passing the Generalized Benford Test

We are now ready to show that $A_{L,T}$ passes the generalized Benford test. The proof will use the following two lemmas.

Lemma 2 *Let S be an irreducible pattern with probability p, and let Z be a Turing machine such that $U(Z,N)$ accepts in time $T(N)$ if and only if $N \in S$.*

There exists a constant C such that if $N \in S$, then there exists a $P \in J_N$ such that

$$\max_{Y \in TM(N)} \; B_N(Z,Y,P) < C\,. \tag{7}$$

Proof. Let $P = \frac{\lfloor pN \rfloor}{N}$. From the definition of irreducible pattern, we have that there exists c such that for all Y,

$$|F_N(Z,Y) - p| < \frac{cK(Y)\sqrt{\log\log Q_N(Z,Y)}}{\sqrt{Q_N(Z,Y)}}\,. \tag{8}$$

Clearly,

$$|P-p| \leq \frac{1}{N} \leq \frac{1}{Q_N(Z,Y)} \leq \frac{1}{\sqrt{Q_N(Z,Y)}} \leq \frac{K(Z)K(Y)\sqrt{\log\log Q_N(Z,Y)}}{\sqrt{Q_N(Z,Y)}}\,. \tag{9}$$

Setting $C = K(Z) + c$, we get

$$|F_N(Z,Y) - P| \leq |F_N(Z,Y) - p| + |P-p| < \frac{CK(Y)\sqrt{\log\log Q_N(Z,Y)}}{\sqrt{Q_N(Z,Y)}}\,, \tag{10}$$

so

$$\frac{|F_N(Z,Y) - P|\sqrt{Q_N(Z,Y)}}{K(Y)\sqrt{\log\log Q_N(Z,Y)}} < C\,. \tag{11}$$

Clearly, $K(Z) < C$, so $B_N(Z,Y,P) > C$ for all Y. Therefore,

$$\max_{Y \in TM(N)} \; B_N(Z,Y,P) < C\,. \tag{12}$$

Lemma 3 *Let S be an irreducible pattern with probability p, and let Z be a Turing machine such that $U(Z,N)$ accepts in time $T(N)$ if and only if $N \in S$.*

For all C, for all $\varepsilon > 0$, for all N sufficiently large, for all $P \in J_N$, if $N \in S$, and

$$\min_{X \in TM(N)} \; B_N(X,Z,P) < C\,, \tag{13}$$

then $|P-p| < \varepsilon$.

Proof. Fix a C and a $\varepsilon > 0$. It suffices to show that for all N sufficiently large, if $N \in S$ and $|P-p| \geq \varepsilon$, then for all $X \in TM(N)$, we have $B_N(X,Z,P) \geq C$.

Observe that since $B_N(X,Z,P) \geq K(X)$, this claim trivially holds when $K(X) \geq C$. Therefore we only have to check the claim for the finitely many Turing machines expressible in fewer than C bits.

Fix an arbitrary X. Since S is an irreducible pattern, there exists a c such that

$$|F_N(X,Z) - p| < \frac{cK(Z)\sqrt{\log\log Q_N(X,Z)}}{\sqrt{Q_N(X,Z)}}\,. \tag{14}$$

We may assume that $S'(X,Z)$ is infinite, since otherwise if we take $N \in S$ large enough, $X \notin TM(N)$. Thus, by taking N sufficiently large, we can get $Q_N(X,Z)$ sufficiently large, and in particular satisfy

$$\frac{\sqrt{Q_N(X,Z)}}{K(Z)\sqrt{\log\log Q_N(X,Z)}}\varepsilon \geq C + c\,. \tag{15}$$

Take $N \in S$ large enough that this holds for each $X \in TM(N)$ with $K(X) < C$, and assume $|P - p| \geq \varepsilon$. By the triangle inequality, we have

$$|F_N(X,Z) - P| \geq |P - p| - |F_N(X,Z) - p| \geq \varepsilon - \frac{cK(Z)\sqrt{\log\log Q_N(X,Z)}}{\sqrt{Q_N(X,Z)}}\,. \tag{16}$$

Therefore

$$\begin{aligned} B_N(X,Z,P) &\geq \frac{\left(\varepsilon - \frac{cK(Z)\sqrt{\log\log Q_N(X,Z)}}{\sqrt{Q_N(X,Z)}}\right)\sqrt{Q_N(X,Z)}}{K(Z)\sqrt{\log\log Q_N(X,Z)}} \\ &= \frac{\sqrt{Q_N(X,Z)}}{K(Z)\sqrt{\log\log Q_N(X,Z)}}\varepsilon - c \geq C\,, \end{aligned} \tag{17}$$

which proves the claim.

Theorem 3 *$A_{L,T}$ passes the generalized Benford test.*

Proof. Let S be an irreducible pattern with probability p. We must show that

$$\lim_{\substack{N\to\infty \\ N\in S}} A_{L,T}(N) = p. \tag{18}$$

Let Z be a Turing machine such that $U(Z,N)$ accepts in time $T(N)$ if and only if $N \in S$.

By considering the case when $X = Z$, Lemma 2 implies that there exists a constant C such that for all N sufficiently large, there exists a $P \in J_N$ such that

$$\max_{Y\in TM(N)} \min_{X\in TM(N)} B_N(X,Y,P) < C. \tag{19}$$

Similarly, using this value of C, and considering the case where $Y = Z$, Lemma 3 implies that for all $\varepsilon > 0$, for all N sufficiently large, for all $P \in J_N$ if $N \in S$, and

$$\max_{Y\in TM(N)} \min_{X\in TM(N)} B_N(X,Y,P) < C, \tag{20}$$

then $|P - p| \leq \varepsilon$.

Combining these, we get that for all $\varepsilon > 0$, for all N sufficiently large, if $N \in S$ and if P is in

$$\arg\min_{P \in J_N} \max_{Y \in TM(N)} \min_{X \in TM(N)} B_N(X, Y, P), \tag{21}$$

then $|P - p| \leq \varepsilon$.

Thus, by Lemma 1, we get that for all $\varepsilon > 0$, for all N sufficiently large, if $N \in S$, then $|A_{L,T}(N) - p| \leq \varepsilon$, so

$$\lim_{\substack{N \to \infty \\ N \in S}} A_{L,T}(N) = p. \tag{22}$$

7 Final Remarks

We identified a new desirable property for logical uncertainty, the generalized Benford test, based on making probability assignments when sequences of logical statements appear pseudorandom. We developed an algorithm with this property. Although the algorithm does not have a practically useful run-time, it demonstrates that it is possible to achieve the desired property in a very general case: we can apply this algorithm to learn patterns in ZFC or other powerful logics, which include essentially any mathematical domains of interest within them.

The main drawback of the approach here is that it does not achieve desirable properties of previous approaches. No attempt is made here to satisfy the probability axioms in the limit as more computing power is used, as in [9, 1]. Integrating with those approaches is an important next step. Nonetheless, we see passing the generalized Benford test alone as a fairly powerful property, as it implies an ability to notice a wide variety of patterns within mathematics.

References

[1] Paul Christiano. *Non-Omniscience, Probabilistic Inference, and Metamathematics*. Tech. rep. 2014–3. Berkeley, CA: Machine Intelligence Research Institute, 2014. URL: `http://intelligence.org/files/Non-Omniscience.pdf`.

[2] Philipp Hennig, Michael A. Osborne, and Mark Girolami. "Probabilistic numerics and uncertainty in computations". In: *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* 471.2179 (2015).

[3] Nels E Beckman and Aditya V Nori. "Probabilistic, modular and scalable inference of typestate specifications". In: *ACM SIGPLAN Notices*. Vol. 46. 6. ACM. 2011, pp. 211–221.

[4] George Polya. *Mathematics and Plausible Reasoning: Patterns of plausible inference*. Vol. 2. Princeton University Press, 1968.

[5] Rohit Parikh. "Knowledge and the Problem of Logical Omniscience." In: *ISMIS*. Vol. 87. 1987, pp. 432–439.

[6] Haim Gaifman. "Reasoning with Limited Resources and Assigning Probabilities to Arithmetical Statements". In: *Synthese* 140.1–2 (2004), pp. 97–119. DOI: 10.1023/B:SYNT.0000029944.99888.a7.

[7] Mikaël Cozic. "Impossible states at work: Logical omniscience and rational choice". In: *Contributions to Economic Analysis* 280 (2006), pp. 47–68.

[8] Joseph Y Halpern and Riccardo Pucella. "Dealing with logical omniscience". In: *Proceedings of the 11th conference on Theoretical aspects of rationality and knowledge*. ACM. 2007, pp. 169–176.

[9] Abram Demski. "Logical Prior Probability". In: *Artificial General Intelligence*. Lecture Notes in Artificial Intelligence 7716. New York: Springer, 2012, pp. 50–59. DOI: 10.1007/978-3-642-35506-6.

[10] Nate Soares and Benja Fallenstein. *Questions of Reasoning Under Logical Uncertainty*. Tech. rep. 2015–1. Berkeley, CA: Machine Intelligence Research Institute, 2015. URL: https://intelligence.org/files/QuestionsLogicalUncertainty.pdf.

[11] Richard T Cox. *Algebra of Probable Inference*. JHU Press, 1961.

[12] Alfred Horn and Alfred Tarski. "Measures in Boolean algebras". In: *Transactions of the American Mathematical Society* 64.3 (1948), pp. 467–467. ISSN: 0002-9947. DOI: 10.1090/S0002-9947-1948-0028922-8. URL: http://www.ams.org/journals/tran/1948-064-03/S0002-9947-1948-0028922-8/S0002-9947-1948-0028922-8.pdf.

[13] Dorothy Maharam. "An algebraic characterization of measure algebras". In: *Annals of Mathematics* 48.1 (1947), pp. 154–167. ISSN: 01960644. DOI: 10.1016/j.annemergmed.2010.11.022.

[14] Jerzy Łoś. "On the axiomatic treatment of probability". In: *Colloquium Mathematicae*. Vol. 2. 3. 1955, pp. 125–137.

[15] Haim Gaifman. "Concerning Measures in First Order Calculi". In: *Israel Journal of Mathematics* 2.1 (1964), pp. 1–18. DOI: 10.1007/BF02759729.

[16] Nils J Nilsson. "Probabilistic logic". In: *Artificial intelligence* 28.1 (1986), pp. 71–87.

[17] Theodore Hailperin et al. "Probability logic." In: *Notre Dame Journal of Formal Logic* 25.3 (1984), pp. 198–212.

[18] Marcus Hutter et al. "Probabilities on sentences in an expressive logic". In: *Journal of Applied Logic* 11.4 (2013), pp. 386–420.

[19] Volodimir G Vovk. "Aggregating strategies". In: *Proc. Third Workshop on Computational Learning Theory*. Morgan Kaufmann. 1990, pp. 371–383.

[20] Marcelo J Weinberger and Erik Ordentlich. "On delayed prediction of individual sequences". In: *Information Theory, IEEE Transactions on* 48.7 (2002), pp. 1959–1976.

[21] L. Pietronero et al. "Explaining the uneven distribution of numbers in nature: the laws of Benford and Zipf". In: *Physica A: Statistical Mechanics and its Applications* 293.1-2 (2001), pp. 297–304. ISSN: 03784371. DOI: 10.1016/S0378-4371(00)00633-6. eprint: 9808305.

[22] KerI Ko. "On the notion of infinite pseudorandom sequences". In: *Theoretical Computer Science* 48 (1986), pp. 9–33. ISSN: 03043975. DOI: 10.1016/0304-3975(86)90081-2.

[23] Rodney G. Downey and Denis R. Hirschfeldt. *Algorithmic randomness and complexity*. Springer Science & Business Media, 2010. ISBN: 9780387955674. DOI: 10.4249/scholarpedia.2574.

[24] Scott Garrabrant et al. *Asymptotic Logical Uncertainty and The Benford Test*. Preprint. 2015. arXiv: 1510.03370 [cs.LG].

[25] F. C. Hennie and R. E. Stearns. "Two-tape simulation of multitape Turing machines". In: *Journal of the ACM* 13.4 (1966), pp. 533–546. ISSN: 00045411. DOI: 10.1145/321356.321362.