Algorithmic Computability and Approximability of Capacity-Achieving Input Distributions

Holger Boche, *Fellow, IEEE*, Rafael F. Schaefer, *Senior Member, IEEE*, and H. Vincent Poor, *Life Fellow, IEEE*

***Abstract*—The capacity of a channel can usually be characterized as a maximization of certain entropic quantities. From a practical point of view it is of primary interest to not only compute the capacity value, but also to find the corresponding optimizer, i.e., the capacity-achieving input distribution. This paper addresses the general question of whether or not it is possible to find algorithms that can compute the optimal input distribution depending on the channel. For this purpose, the concept of Turing machines is used which provides the fundamental performance limits of digital computers and therewith fully specifies tasks are algorithmically feasible in principle. It is shown for discrete memoryless channels that it is impossible to algorithmically compute the capacity-achieving input distribution, where the channel is given as an input to the algorithm (or Turing machine). Finally, it is further shown that it is even impossible to algorithmically approximate these input distributions.**

***Index Terms*—Capacity-achieving input distribution, Turing machine, computability, approximability.**

I. Introduction

The capacity of a channel describes the maximum rate at which a sender can reliably transmit a message over a noisy channel to a receiver. Accordingly, the capacity is a function of the channel and is usually expressed by entropic quantities that are maximized over all possible input distributions. To this end, a (numerical) evaluation of the capacity and a characterization of the optimal input distribution that maximizes the capacity expression are important and common tasks in information and communication theory. To date, for discrete memoryless channels (DMCs) no closed form solution for the optimal input distribution as a function of the channel is known. Therefore, several approaches have been proposed to algorithmically compute the capacity and also (implicitly) the corresponding optimizer. This is an interesting and challenging task which can be seen already for the binary symmetric channel whose capacity is a transcendental number[1] in general (see also the appendix for a detailed discussion on this). Thus, an exact computation of the capacity value is not possible on a digital computer as any practical algorithm must stop after a finite number of computation steps and, therefore, only an approximation of the capacity value is possible.

A famous iterative algorithm for the computation of the capacity of an arbitrary DMC was independently proposed in 1972 by Arimoto [1] and Blahut [2], where the latter further presented a corresponding algorithm for the computation of the rate-distortion function. This iterative algorithm is now referred to as the *Blahut-Arimoto algorithm*. It was further studied by Csiszár [3] and later generalized by Csiszár and Tusnády [4]. The Blahut-Arimoto algorithm also appears in introductory textbooks on information theory such as [5] and [6]. Since then, the Blahut-Arimoto algorithm has been extensively studied and extended to various scenarios, cf. for example [7–14].

Blahut motivated his studies in [2] by the desire to use digital computers, which were becoming more and more powerful at this time, for the numerical computation of the capacity of DMCs. Since the seminal works [1] and [2], digital computers have been extensively used in information and communication theory to simulate and evaluate the performance of communication systems. Not surprisingly, higher-layer network simulations on high performance computers became a commonly used approach for the design of practical systems. A critical discussion on this trend is given in [15].

In this paper, we address the issue of computing the optimal input distribution from a fundamental algorithmic point of view by using the concept of a *Turing machine* [16–18] and the corresponding *computability framework*. The Turing machine is a mathematical model of an abstract machine that manipulates symbols on a strip of tape according to certain given rules. It can simulate any given algorithm and therewith provides a simple but very powerful model of computation. Turing machines have no limitations on computational complexity, unlimited computing capacity and storage, and execute programs completely error-free. They are further equivalent to the von Neumann-architecture without hardware limitations

This work of H. Boche was supported in part by the German Federal Ministry of Education and Research (BMBF) within the national initiative on 6G Communication Systems through the research hub *6G-life* under Grant 16KISK002, within the national initiative on *Post Shannon Communication (NewCom)* under Grant 16KIS1003K, and the project *Hardware Platforms and Computing Models for Neuromorphic Computing (NeuroCM)* under Grant 16ME0442. It has further received funding by the Bavarian Ministry of Economic Affairs, Regional Development and Energy as part of the project *6G Future Lab Bavaria* as well as by the German Research Foundation (DFG) within Germany's Excellence Strategy EXC-2092 – 390781972. This work of R. F. Schaefer was supported in part by the BMBF within NewCom under Grant 16KIS1004 and in part by the DFG under Grant SCHA 1944/6-1. This work of H. V. Poor was supported by the U.S. National Science Foundation under Grant CCF-1908308.

Holger Boche is with the Institute of Theoretical Information Technology, Technical University of Munich, 80290 Munich, Germany, the BMBF Research Hub 6G-life, 80290 Munich, Germany, and the Excellence Cluster Cyber Security in the Age of Large-Scale Adversaries (CASA), Ruhr University Bochum, 44801 Bochum, Germany (email: boche@tum.de).

Rafael F. Schaefer is with the Chair of Communications Engineering and Security, and the Center for Sensor Systems (ZESS), University of Siegen, 57068 Siegen, Germany (email: rafael.schaefer@uni-siegen.de).

H. Vincent Poor is with the Department of Electrical and Computer Engineering, Princeton University, Princeton, NJ 08544, USA (email: poor@princeton.edu).

[1] An *algebraic number* is a number that is a root of a non-zero polynomial with integer coefficients. A *transcendental number* is a number that is not algebraic, i.e., it is not a root of any non-zero integer polynomial.

and the theory of recursive functions, cf. also [19–23]. Accordingly, Turing machines provide fundamental performance limits for today's digital computers and are the ideal concept to study whether or not such computation tasks can be done algorithmically in principle.

Communication from a computability or algorithmic point of view has attracted some attention recently. In [24] the computability of the capacity functions of the wiretap channel under channel uncertainty and adversarial attacks is studied. The computability of the capacity of finite state channels is studied in [25] and of non-i.i.d. channels in [26]. These works have in common that they study capacity functions of various communication scenarios and analyze the algorithmic computability of the capacity function itself. While for DMCs the capacity function is a computable continuous function and therewith indeed algorithmically computable [27, 28], this is no longer the case for certain multi-user scenarios or channels with memory. However, they do not consider the computation of the optimal input distributions which, to the best of our knowledge, have not been studied so far from a fundamental algorithmic point of view. In addition, even if the capacity is computable, it is still not clear whether or not the corresponding optimal input distributions can be algorithmically computed.

We consider finite input and output alphabets. Due to the properties of the mutual information, the set of capacity-achieving input distributions is mathematically well defined for every DMC and so are all functions that map every channel to a corresponding capacity-achieving input distribution. A practically relevant question is now whether or not these functions are also algorithmically well defined. With this we mean whether or not it is possible to find at least one function that can be implemented by an algorithm (or Turing machine). This is equivalent to the question of whether or not a Turing machine exists that gets a computable channel as input and subsequently computes an optimal input distribution of this channel.

In this paper, we give a negative answer to the question above by showing that it is in general impossible to find an algorithm (or Turing machine) that is able to compute the optimal input distribution when the channel is given as an input. To this end, we first introduce the computability framework based on Turing machines in Section II. The communication system model and the precise problem formulation are subsequently introduced in Section III. We show that all functions that map channels to their corresponding optimal input distributions are not Banach-Mazur computable and therewith also not Turing computable. As a consequence, there is no algorithm (or Turing machine) that is able to compute the optimizer, i.e., the capacity-achieving input distribution. Subsequently, it is shown that it is further not even possible to algorithmically approximate the optimizer, i.e., the capacity-achieving input distribution, within a given tolerated error. The corresponding proofs are given in Section IV for the non-computability of the optimizer and in Section V for the non-approximability. Finally, a conclusion is given in Section VI.

Notation

Discrete random variables are denoted by capital letters and their realizations and ranges by lower case and calligraphic letters, respectively; all logarithms and information quantities are taken to the base 2; $\mathbb{N}$, $\mathbb{Q}$, and $\mathbb{R}$ are the sets of non-negative integers, rational numbers, and real numbers; $\mathcal{P}(\mathcal{X})$ denotes the set of all probability distributions on $\mathcal{X}$ and $\mathcal{CH}(\mathcal{X};\mathcal{Y})$ denotes the set of all stochastic matrices (channels) $\mathcal{X} \to \mathcal{P}(\mathcal{Y})$; the binary entropy is denoted by $h_2(p) = -p\log p - (1-p)\log(1-p)$ and $I(X;Y)$ denotes the mutual information between the input X and the output Y which we interchangeably also write as $I(p,W)$ to emphasize the dependency on the input distribution $p \in \mathcal{P}(\mathcal{X})$ and the channel $W \in \mathcal{CH}(\mathcal{X};\mathcal{Y})$; the ℓ_1-norm is denoted by $\|\cdot\|_{\ell_1}$.

II. Computability Framework

We first introduce the computability framework based on Turing machines which provides the needed background. Turing machines are extremely powerful compared to state-of-the-art digital signal processing (DSP) and field gate programmable array (FPGA) platforms and even current supercomputers. It is the most general computing model and is even capable of performing arbitrary exhaustive search tasks on arbitrary large but finite structures. The complexity can even grow faster than double-exponentially with the set of parameters of the underlying communication system (such as time, frequencies, transmit power, modulation scheme, number of antennas, etc.).

In what follows, we need some basic definitions and concepts of computability which are briefly reviewed. The concept of computability and computable real numbers was first introduced by Turing in [16] and [17].

Recursive functions $f : \mathbb{N} \to \mathbb{N}$ map natural numbers into natural numbers and are exactly those functions that are computable by a Turing machine. They are the smallest class of partial functions that includes the primitive functions (i.e., constant function, successor function, and projection function) and is further closed under composition, primitive recursion, and minimization. For a detailed introduction, we refer the reader to [29] and [27]. With this, we call a sequence of rational numbers $(r_n)_{n\in\mathbb{N}}$ a *computable sequence* if there exist recursive functions $a, b, s : \mathbb{N} \to \mathbb{N}$ with $b(n) \neq 0$ for all $n \in \mathbb{N}$ and

$$r_n = (-1)^{s(n)} \frac{a(n)}{b(n)}, \qquad n \in \mathbb{N}; \tag{1}$$

cf. [29, Def. 2.1 and 2.2] for a detailed treatment. A real number x is said to be computable if there exists a computable sequence of rational numbers $(r_n)_{n\in\mathbb{N}}$ and a recursive function φ such that we have for all $M \in \mathbb{N}$

$$|x - r_n| < 2^{-M} \tag{2}$$

for all $n \geq \varphi(M)$. Thus, the computable real x is represented by the pair $((r_n)_{n\in\mathbb{N}}, \varphi)$. Note that a computable real number usually has multiple different representations. For example, there are multiple algorithms known for the computation of $\frac{1}{\pi}$ or e^{-1}. This form of convergence (2) with a computable

control of the approximation error is called *effective convergence*. Note that if a computable sequence of computable real numbers $(r_n)_{n\in\mathbb{N}}$ converges effectively to a limit x, then x is a computable real number, cf. [27]. Furthermore, the set $\mathbb{R}_c$ of all computable real numbers is closed for addition, subtraction, multiplication, and division (excluding the division by zero). We denote the set of computable real numbers by $\mathbb{R}_c$. Based on this, we define the set of computable probability distributions $\mathcal{P}_c(\mathcal{X})$ as the set of all probability distributions $P_X \in \mathcal{P}(\mathcal{X})$ such that $P_X(x) \in \mathbb{R}_c$ for all $x \in \mathcal{X}$. Further, let $\mathcal{CH}_c(\mathcal{X};\mathcal{Y})$ be the set of all computable channels, i.e., for a channel $W : \mathcal{X} \to \mathcal{P}(\mathcal{Y})$ we have $W(\cdot|x) \in \mathcal{P}_c(\mathcal{Y})$ for every $x \in \mathcal{X}$.

Definition 1. A function $f : \mathbb{R}_c \to \mathbb{R}_c$ is called *Borel-Turing computable* if there exists an algorithm or Turing machine $\mathfrak{T}_f$ such that $\mathfrak{T}_f$ obtains for every x an arbitrary representation $((r_n)_{n\in\mathbb{N}}, \varphi)$ for it as input and then computes a representation $((\hat{r}_n)_{n\in\mathbb{N}}, \hat{\varphi})$ for $f(x)$.

Remark 1. Borel-Turing computability characterizes exactly the behavior that is expected when functions are simulated and evaluated on digital hardware platforms. A program for the computation of $f(x)$ must receive a representation $((r_n)_{n\in\mathbb{N}}, \varphi)$ for the input x. Based on this, the program computes the representation $((\hat{r}_n)_{n\in\mathbb{N}}, \hat{\varphi})$ for $f(x)$. This means that if $f(x)$ needs to be computed with a tolerated approximation error of $\frac{1}{2^M}$, then it is sufficient to compute the rational number $\hat{r}_{\hat{\varphi}(M)}$ and the corresponding Turing machine outputs $\hat{r}_{\hat{\varphi}(M)}$. For example, this is done and further discussed for the function $f(x) = e^{-x}$, $x \in [0,1]$, $x \in \mathbb{R}_c$ in Appendix A.

Remark 2. A practical digital hardware platform and also a Turing machine must stop after finitely many computation steps when computing a value of a function. Thus, the computed value of the function must be a rational number. As a consequence, a Turing machine can only compute rational numbers *exactly*. However, it is important to note that in information and communication theory, the relevant information-theoretic functions are in general not exactly computable even for rational channel and system parameters. For example, already for $|\mathcal{X}| = 2$ and rational probability distribution $p \in \mathcal{P}(\mathcal{X})$, $p \neq \left(\frac{1}{2}, \frac{1}{2}\right)$, the corresponding binary entropy $h_2(p)$ is a transcendental number and therewith not exactly computable. Even if this would be done symbolically with algebraic numbers, the binary entropy would not be computable. As a consequence, already for the binary symmetric channel (BSC) with rational crossover probability $\epsilon \in (0, \frac{1}{2}) \cap \mathbb{Q}$, the capacity $C_{\text{BSC}}(\epsilon) = 1 - h_2(\epsilon)$ is a transcendental number and therewith an exact computation of the capacity is not possible. A proof for this statement is given in Appendix B for completeness.

There are also weaker forms of computability including *Banach-Mazur computability*. In particular, Borel-Turing computability implies Banach-Mazur computability, but not vice versa. For an overview of the logical relations between different notions of computability we refer to [19] and, for example, the introductory textbook [18].

Definition 2. A function $f : \mathbb{R}_c \to \mathbb{R}_c$ is called *Banach-Mazur computable* if f maps any given computable sequence $(x_n)_{n\in\mathbb{N}}$ of computable real numbers into a computable sequence $(f(x_n))_{n\in\mathbb{N}}$ of computable real numbers.

We further need the concepts of a recursive set and a recursively enumerable set as, for example, defined in [29].

Definition 3. A set $\mathcal{A} \subset \mathbb{N}$ is called *recursive* if there exists a computable function f such that $f(x) = 1$ if $x \in \mathcal{A}$ and $f(x) = 0$ if $x \notin \mathcal{A}$.

Definition 4. A set $\mathcal{A} \subset \mathbb{N}$ is *recursively enumerable* if there exists a recursive function whose range is exactly $\mathcal{A}$.

We have the following properties which will be crucial later for proving the desired results; cf. also [29] for further details.

- $\mathcal{A}$ is recursive is equivalent to $\mathcal{A}$ is recursively enumerable and $\mathcal{A}^c$ is recursively enumerable.
- There exist recursively enumerable sets $\mathcal{A} \subset \mathbb{N}$ that are not recursive, i.e., $\mathcal{A}^c$ is not recursively enumerable. This means there are no computable, i.e., recursive, functions $f : \mathbb{N} \to \mathcal{A}^c$ where for each $m \in \mathcal{A}^c$ there exists an x with $f(x) = m$.

III. Problem Formulation and Main Results

Here, we introduce the communication scenario of interest and present the main problem and results of this work.

A. Communication System Model

We consider a point-to-point channel with one transmitter and one receiver which defines the most basic communication scenario. Let $\mathcal{X}$ and $\mathcal{Y}$ be finite input and output alphabets. Then the channel is given by a stochastic matrix $W : \mathcal{X} \to \mathcal{P}(\mathcal{Y})$ which we also equivalently write as $W \in \mathcal{CH}(\mathcal{X};\mathcal{Y})$. The corresponding DMC is then given by $W^n(y^n|x^n) := \prod_{i=1}^n W(y_i|x_i)$ for all $x^n \in \mathcal{X}^n$ and $y^n \in \mathcal{Y}^n$.

Definition 5. An (M_n, E_n, D_n)*-code* $\mathcal{C}_n(W)$ of blocklength $n \in \mathbb{N}$ for the DMC $W \in \mathcal{CH}(\mathcal{X};\mathcal{Y})$ consists of an encoder $E_n : \mathcal{M}_n \to \mathcal{X}^n$ at the transmitter with a set of messages $\mathcal{M}_n := \{1, ..., M_n\}$ and a decoder $D_n : \mathcal{Y}^n \to \mathcal{M}_n$ at the receiver.

The transmitted codeword needs to be decoded reliably at the receiver. To model this requirement, we define the *average probability of error* as

$$\bar{e}_n := \frac{1}{|\mathcal{M}_n|} \sum_{m\in\mathcal{M}_n} \sum_{y^n : D_n(y^n)\neq m} W^n(y^n|x_m^n)$$

and the *maximum probability of error* as

$$e_{\max,n} := \max_{m\in\mathcal{M}_n} \sum_{y^n : D_n(y^n)\neq m} W^n(y^n|x_m^n)$$

with $x_m^n = E_n(m)$ the codeword for message $m \in \mathcal{M}_n$.

Definition 6. A rate $R > 0$ is called *achievable* for the DMC W if there exists a sequence $(\mathcal{C}_n(W))_{n\in\mathbb{N}}$ of (M_n, E_n, D_n)-codes such that we have $\frac{1}{n}\log M_n \geq R$ and $\bar{e}_n \leq \epsilon_n$ (or $e_{\max,n} \leq \epsilon_n$, respectively) with $\epsilon_n \to 0$ as $n \to \infty$. The

capacity $C(W)$ of the DMC W is given by the supremum of all achievable rates R.

The capacity of the DMC has been established and goes back to the seminal work of Shannon [30].

Theorem 1. The capacity $C(W)$ of the DMC W under both the average and maximum error criteria is

$$C(W) = \max_X I(X;Y) = \max_{p \in \mathcal{P}(\mathcal{X})} I(p,W). \tag{3}$$

The capacity of a channel characterizes the maximum transmission rate at which the users can reliably communicate with vanishing probability of error. Note that for DMCs, there is no difference in the capacity whether the average error or the maximum error criterion is considered.

Remark 3. Capacity expressions such as (3) for the point-to-point channel have further been established for various multi-user communication scenarios, cf. for example [31] and references therein. They all have in common that these are characterized by entropic quantities.

B. Blahut-Arimoto Algorithm

The Blahut-Arimoto algorithm as initially proposed in [1] and [2] tackles the problem of numerically computing the capacity of DMCs with finite input and output alphabets. This algorithm is an alternating optimization algorithm, which has become a standard technique of convex optimization. It has the advantage that it exploits the properties of the mutual information to obtain a simple method to compute the capacity.

For a DMC W, the algorithm computes the following two quantities at the n-th iteration:

1) an input distribution $p_n = p_n(W)$
2) an approximation to the capacity given by the mutual information $I(p_n, W)$ for this input distribution.

This means the algorithm computes a sequence $p_0(W)$, $I(p_0, W)$, $p_1(W)$, $I(p_1, W)$, ... , $p_n(W)$, $I(p_n, W)$, ... where each element in the sequence is a function of the previous ones except the initial input distribution $p_0(W)$ which is arbitrarily chosen.

For the sequence $p_0(W)$, $p_1(W)$, ... it is shown in [1–3] that it converges to an optimizer, i.e., to an optimal input distribution. First, the existence of a limit $p_* = p_*(W) \in \mathcal{P}(\mathcal{X})$ of this sequence is shown by the Bolzano–Weierstrass theorem, cf. for example [32]. Subsequently, it is shown that this limit must be an optimal input distribution, i.e., $p_* \in \mathcal{P}_{\text{opt}}(W)$ with

$$\mathcal{P}_{\text{opt}}(W) = \big\{p \in \mathcal{P}(\mathcal{X}) : I(p,W) = C(W)\big\} \tag{4}$$

the set of optimal input distributions. The Heine-Borel theorem is a simple technique to show the existence of solutions of certain problems, but, in general, it does not provide an algorithm to compute this solution; in this case the optimal input distribution as a function of the channel.

For the capacity, a stopping criterion is provided, i.e., we can choose a certain approximation error $\frac{1}{2^M} > 0$, $M \in \mathbb{N}$, and the algorithm stops if this tolerated error is satisfied so that the computed value $I(p_n, W)$ is within this error to the actual capacity $C(W)$, i.e.,

$$\big|C(W) - I(p_n, W)\big| < \frac{1}{2^M}.$$

On the other hand, for the optimizer, i.e., the optimal input distribution, a stop criterion has not been given in [1–3], i.e., we cannot control when the algorithm should stop for a given maximum tolerable error. Such a stopping criterion could similarly be defined, e.g., when

$$\big\|p_* - p_n(W)\big\|_{\ell_1} < \frac{1}{2^M}$$

with $p_* \in \mathcal{P}_{\text{opt}}(W)$ a capacity-achieving input distribution is satisfied and further a computable upper bound for the speed of convergence is given. Surprisingly, to date such a stopping criterion has not been found. In particular, our results even show that such a stopping criterion cannot exist! We will come back to this issue in more detail in the following subsection.

In fact, both seminal papers [1] and [2] do not only aim at computing the capacity, but also propose an algorithm for the computation of a sequence of input distributions $p_n \in \mathcal{P}(\mathcal{X})$ and study the convergence to a maximum $p_* \in \mathcal{P}_{\text{opt}}(W)$ for a fixed channel W, i.e.,

$$I(p_*, W) = C(W) = \max_{p \in \mathcal{P}(\mathcal{X})} I(p,W).$$

They state that a suitable subsequence $(p_{n_l})_{l \in \mathbb{N}}$ converges to an optimizer, but without providing a stopping criterion. That this is problematic has been realized afterwards by Csiszár who explicitly states in [3] that there is no stopping criterion for the computation of the optimizer.

C. Computability of an Optimal Input Distribution

The capacity $C(W) = \max_{p \in \mathcal{P}(\mathcal{X})} I(p,W)$ of the DMC W, cf. (3), is given by a maximization problem, where the mutual information $I(p,W)$ is maximized over all possible input distributions $p \in \mathcal{P}(\mathcal{X})$. Since $I(p,W)$ is continuous in (p,W), concave in the input distribution p, and convex in the channel W, there exists for every channel $W \in \mathcal{CH}(\mathcal{X};\mathcal{Y})$ at least one optimal input distribution $p_*(W) \in \mathcal{P}_{\text{opt}}(W)$. Note that the set $\mathcal{P}_{\text{opt}}(W)$ is a convex set for each channel W. Now, we can choose for every channel $W \in \mathcal{CH}(\mathcal{X};\mathcal{Y})$ such a capacity-achieving input distribution $p_* = p_*(W)$. Then $F(W) = p_*(W)$ is a mathematically well defined function of the form

$$F : \mathcal{CH}(\mathcal{X};\mathcal{Y}) \to \mathcal{P}(\mathcal{X}) \tag{5}$$

which maps every channel to an optimal input distribution for this channel. We call F an optimal assignment function and denote by $\mathcal{M}_{\text{opt}}(\mathcal{X};\mathcal{Y})$ the set of all these functions. The set $\mathcal{M}_{\text{opt}}(\mathcal{X};\mathcal{Y})$ is of crucial practical importance and, in particular, it would be interesting to find functions $F \in \mathcal{M}_{\text{opt}}(\mathcal{X};\mathcal{Y})$ that can be described algorithmically. Note that in general, this function F does not need to be unique and there can be infinitely many such functions. Further, for computable channels $W \in \mathcal{CH}_c(\mathcal{X};\mathcal{Y})$ we always have $F(W) \in \mathcal{P}_c(\mathcal{X})$.

Remark 4. From a practical point of view it is interesting to understand whether or not there exists a function F

with $F(W) \in \mathcal{P}_{\text{opt}}(W)$ for all $W \in \mathcal{CH}_c(\mathcal{X};\mathcal{Y})$ that is Borel-Turing computable. Since exactly in this case there is an algorithm (or Turing machine) that takes the channel $W \in \mathcal{CH}_c(\mathcal{X};\mathcal{Y})$ as an input and computes a corresponding capacity-achieving input distribution $F(W) = p_*(W) \in \mathcal{P}_{\text{opt}}(W)$. It is clear that we consider only computable channels $W \in \mathcal{CH}_c(\mathcal{X};\mathcal{Y})$ as inputs for the Turing machine as it can operate work only with such inputs. More specifically, for $W \in \mathcal{CH}_c(\mathcal{X};\mathcal{Y})$ such a Turing machine gets an arbitrary representation of W as input, i.e., $W(y|x)$ is given by a representation $((r_n(x,y))_{n\in\mathbb{N}}, \varphi^{(x,y)})$ for all $x \in \mathcal{X}$, $y \in \mathcal{Y}$. This means that for all $x \in \mathcal{X}$, $y \in \mathcal{Y}$ we have for all $N \in \mathbb{N}$

$$\big|W(y|x) - r_n(x,y)\big| < \frac{1}{2^N}$$

for all $n \geq \varphi^{(x,y)}(N)$. As a result, the Turing machine computes a representation of $F(W) \in \mathcal{P}_{\text{opt}}(W)$, i.e., $((r_n^*(x))_{n\in\mathbb{N}}, \varphi^{(*,x)})$ is a representation of $p_*(x)$, $x \in \mathcal{X}$, with $F(W) = p_* = \big(p_*(1), \ldots, p_*(|\mathcal{X}|)\big)$. Thus, for all $x \in \mathcal{X}$ it holds that for all $N \in \mathbb{N}$

$$\big|p_*(x) - r_n^*(x)\big| < \frac{1}{2^N} \tag{6}$$

for all $n \geq \varphi^{(*,x)}(N)$.

Accordingly, in the following we will address this question in detail and study whether or not it is possible to find such a Turing machine that computes a capacity-achieving input distribution for a given channel.

Question 1: Let $\mathcal{X}$ and $\mathcal{Y}$ be finite input and output alphabets. Is there an algorithm (or Turing machine) $\mathfrak{T}$ that takes an arbitrary representation of $W \in \mathcal{CH}_c(\mathcal{X};\mathcal{Y})$ as an input and computes a description of $p_*(W) \in \mathcal{P}_{\text{opt}}(W)$?

Remark 5. Question 1 formalizes exactly what we would require from an algorithmic construction of optimal input distributions on digital hardware platforms. From a practical point of view, a simulation on a digital hardware must stop after a finite number of computations. Usually, it should stop if for $W \in \mathcal{CH}_c(\mathcal{X};\mathcal{Y})$ the computed approximation of an input distribution $p_*(W) \in \mathcal{P}_{\text{opt}}(W)$ satisfies a given but fixed approximation error. This constraint on the approximation error is exactly modeled by the representation of $p_*(W)$. If the representation $((r_n^*(x))_{n\in\mathbb{N}}, \varphi^{(*,x)})$, $x \in \mathcal{X}$, of $p_*(W)$ has been computed for a tolerated error $\frac{1}{2^N}$ and r being the smallest natural number such that $2^r > |\mathcal{X}|$, then the approximation process can be stopped after $N^* = \max_{x\in\mathcal{X}} \varphi^{(*,x)}(N+r)$ steps, since we have

$$\sum_{x\in\mathcal{X}} \big|p_*(x) - r_{N^*}^*(x)\big| < \sum_{x\in\mathcal{X}} \frac{1}{2^{N+r}} = \frac{|\mathcal{X}|}{2^{N+r}} < \frac{1}{2^N}.$$

This would provide us a stopping criterion as discussed in Section III-B for the Blahut-Arimoto algorithm.

Now we can state the following result which provides a negative answer to Question 1 above.

Theorem 2. Let $\mathcal{X}$ and $\mathcal{Y}$ be arbitrary but finite alphabets with $|\mathcal{X}| \geq 3$ and $|\mathcal{Y}| \geq 2$. Then there is no function $F \in \mathcal{M}_{opt}(\mathcal{X};\mathcal{Y})$ that is Banach-Mazur computable.

Proof: The proof is given in Section IV. ∎

From this, we can immediately conclude the following.

Corollary 1. There is no Turing machine $\mathfrak{T}$ that takes a channel $W \in \mathcal{CH}_c(\mathcal{X};\mathcal{Y})$ as an input and computes an optimal input distribution $p \in \mathcal{P}_{opt}(W)$ for this channel.

Proof: If such a Turing machine would exist, then the corresponding function F would be Banach-Mazur computable. This is a contradiction to Theorem 2 so that such a Turing machine cannot exist. ∎

Remark 6. This shows that such a Turing machine cannot exist providing a negative answer to Question 1 above. As a consequence, this means also that a function F as in (5) cannot exist for which $F(W)$ can "easily" be computed for W. In particular, this excludes the possibility to find a function F that provides a "closed form solution", since this would be then Turing computable and therewith algorithmically constructable, cf. also [33, 34].

Remark 7. It is of interest to discuss the Blahut-Arimoto algorithm taking the result in Theorem 2 into account. This algorithm computes for each channel $W \in \mathcal{CH}_c(\mathcal{X};\mathcal{Y})$ an optimal input distribution $p_*(W) \in \mathcal{P}_{\text{opt}}(W)$ and a corresponding sequence $(p_n)_{n\in\mathbb{N}}$ for the representation of $p_*(W)$. The second crucial ingredient of the representation of $p_*(W)$ is a stopping criterion for the computation of the sequence $(p_n)_{n\in\mathbb{N}}$ for a given approximation error $\frac{1}{2^N}$. Such a stopping criterion is not provided by the Blahut-Arimoto algorithm. This was already criticized by Csiszár in [3]. Our Theorem 2 shows now that such a computable stopping criterion as a function of the representation of the channel cannot exist!

The statement on the impossibility of the algorithmic solvability is closely connected to the underlying hardware platform (Turing machine) and therewith, equivalently, to the admissible programming languages (Turing complete) and also the admissible signal processing operations. Note that for other computing platforms (such as neuromorphic or quantum computing platforms) this statement need not be the case. However, whenever simulations are done in the broad area of information theory, communication theory, or signal processing, these are done on digital hardware platforms for which Turing machines provide the underlying computing framework.

Remark 8. It is helpful and very interesting to gain further intuition and insight into the non-computability by Turing machines and other potential computing platforms. For example, it has been a long-standing open problem to describe the roots of polynomials by radicals as a function of the coefficients of the polynomial. To this end, Galois showed this is not possible in general for polynomials of the order 5 or higher [35]. This means that the roots of polynomials of order 5 or higher cannot be expressed as a "closed form solution" by radicals; see [35] and further discussions in [33, 34]. On the other hand, from the complex analysis there are algorithms known that are able

to approximate these roots. This shows that the "computing theory of radicals" is not sufficient for the computation of the roots of polynomials of order 5 or higher, but other techniques from complex analysis enable the approximation thereof.

D. Approximability of an Optimal Input Distribution

Above we have shown that it is impossible to algorithmically construct optimal, i.e., capacity-achieving, input distributions. Consequently, we are now interested to understand whether or not it is at least possible to algorithmically approximate such distributions.

We have seen that all functions $F : \mathcal{CH}_c(\mathcal{X};\mathcal{Y}) \to \mathcal{P}(\mathcal{X})$ with $F(W) \in \mathcal{P}_{\text{opt}}(W)$ for all $W \in \mathcal{CH}_c(\mathcal{X};\mathcal{Y})$ are not Banach-Mazur computable and therewith also not Borel-Turing computable. The question is now whether or not we can instead solve this problem approximately, i.e., does there exist a computable sequence of Borel-Turing computable functions F_n, $n \in \mathbb{N}$, with $F_n : \mathcal{CH}_c(\mathcal{X};\mathcal{Y}) \to \mathcal{P}_c(\mathcal{X})$, $n \in \mathbb{N}$, such that for all $W \in \mathcal{CH}_c(\mathcal{X};\mathcal{Y})$ for a suitable function F with $F(W) \in \mathcal{P}_{\text{opt}}(W)$ for all $W \in \mathcal{CH}_c(\mathcal{X};\mathcal{Y})$ we always have

$$\big\|F(W) - F_n(W)\big\|_{\ell_1} < \frac{1}{2^n}.$$

This is equivalent to the question of if there exists a Turing machine $\mathfrak{T}$ that takes an arbitrary representation of $W \in \mathcal{CH}_c(\mathcal{X};\mathcal{Y})$ and $n \in \mathbb{N}$ as inputs and then computes for W and n a representation for $p_n(W) \in \mathcal{P}(\mathcal{X})$ such that

$$\big\|F(W) - p_n(W)\big\|_{\ell_1} < \frac{1}{2^n}. \tag{7}$$

And this is equivalent to the question of whether or not it is possible to find a Turing machine $\mathfrak{T}$ with the following properties: $\mathfrak{T}$ takes the channel and natural numbers as inputs and computes a description of an input distribution. This input distribution must satisfy the following: for all $W \in \mathcal{CH}_c(\mathcal{X};\mathcal{Y})$ and all $n \in \mathbb{N}$ the Turing machine must compute for every description for W a description of $p_n(W)$ such that for a suitable $p_*(W) \in \mathcal{P}_{\text{opt}}(W)$ it always holds

$$\big\|p_*(W) - p_n(W)\big\|_{\ell_1} < \frac{1}{2^n}.$$

The input n of this Turing machine $\mathfrak{T}$ could enable the algorithmic approximation of the optimal input distribution.

A negative answer can be immediately given to this question based on the results obtained above, since a function F must be Borel-Turing computable, see also [25]. We can formalize the following question.

Question 2: Let $\mathcal{X}$ and $\mathcal{Y}$ be finite input and output alphabets with $|\mathcal{X}| \geq 3$ and $|\mathcal{Y}| \geq 2$. Is it possible to approximate a function $F \in \mathcal{M}_{\text{opt}}(\mathcal{X};\mathcal{Y})$ by computable functions. Is there a function $F \in \mathcal{M}_{\text{opt}}(\mathcal{X};\mathcal{Y})$ and a computable function F_1 such that

$$\sup_{W \in \mathcal{CH}_c(\mathcal{X};\mathcal{Y})} \big\|F(W) - F_1(W)\big\|_{\ell_1} < \frac{1}{2}.$$

Remark 9. With this question we ask whether or not the previous condition (7) as the supremum can be satisfied for the trivial case $n = 1$.

Theorem 3. Let $\mathcal{X}$ and $\mathcal{Y}$ be arbitrary but finite alphabets with $|\mathcal{X}| \geq 3$ and $|\mathcal{Y}| \geq 2$. Let $F \in \mathcal{M}_{opt}(\mathcal{X};\mathcal{Y})$ be an arbitrary function and let F_1 be another arbitrary function with

$$\sup_{W \in \mathcal{CH}_c(\mathcal{X};\mathcal{Y})} \big\|F(W) - F_1(W)\big\|_{\ell_1} = \alpha < \frac{1}{2}.$$

Then F_1 is not Banach-Mazur computable.

Proof: The proof is given in Section V. ∎

From this we immediately obtain the following result.

Corollary 2. Let $F \in \mathcal{M}_{opt}(\mathcal{X};\mathcal{Y})$ be an arbitrary function. For $\alpha < \frac{1}{2}$ arbitrary, there exists no Turing machine $\mathfrak{T}_*$ such that for all $W \in \mathcal{CH}_c(\mathcal{X};\mathcal{Y})$,

$$\big\|F(W) - \mathfrak{T}_*(W)\big\|_{\ell_1} \leq \alpha$$

is true.

Proof: If such a function $F \in \mathcal{M}_{\text{opt}}(\mathcal{X};\mathcal{Y})$ would exist for which we can find a Turing machine $\mathfrak{T}_*$ with $\hat{\alpha} < \frac{1}{2}$, then $F_1(W) = \mathfrak{T}_*(W)$, $W \in \mathcal{CH}_c(\mathcal{X};\mathcal{Y})$, would be Banach-Mazur computable. ∎

As a consequence, we can further conclude the following.

Corollary 3. The approximation problem stated in Question 2 is not solvable.

Proof: Already for $n = 2$ this is not possible. ∎

IV. Non-Computability of the Optimizer

Before we present the proof of Theorem 2, we first need to define and discuss specific channels and their optimal input distributions.

A. Preliminary Considerations

Let $\mathcal{X}$ and $\mathcal{Y}$ be arbitrary but finite alphabets with $|\mathcal{X}| = 3$ and $|\mathcal{Y}| = 2$. We define the channel

$$W_* = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{pmatrix} \tag{8}$$

and further consider the channels

$$W_{1,\mu} = \begin{pmatrix} 1 & 0 & \mu \\ 0 & 1 & 1-\mu \end{pmatrix} \quad \text{and} \quad W_{2,\mu} = \begin{pmatrix} 1 & \mu & 0 \\ 0 & 1-\mu & 1 \end{pmatrix}$$

for $\mu \in (0,1)$. We define the distance between two channels $W_1, W_2 \in \mathcal{CH}(\mathcal{X};\mathcal{Y})$ based on the total variation distance as

$$D(W_1, W_2) \coloneqq \max_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} \big|W_1(y|x) - W_2(y|x)\big|$$

and observe that

$$\lim_{\mu \to 0} D(W_*, W_{1,\mu}) = \lim_{\mu \to 0} D(W_*, W_{2,\mu}) = 0.$$

We consider the set

$$\mathcal{P}_1 = \Big\{p = (p_1, p_2, p_3) \in \mathcal{P}(\mathcal{X}) : p_1 = \frac{1}{2} \text{ and } p_2 + p_3 = \frac{1}{2}\Big\}.$$

Then we have

$$\max_{p\in\mathcal{P}(\mathcal{X})} I(p,W_*) = 1 = I(p_*,W_*)$$

with $p_* \in \mathcal{P}_1$ arbitrary. This means $\mathcal{P}_1$ is the set of all maximizing input distributions for the channel W_*, since

$$\begin{aligned} I(p,W_*) &= p_1\cdot 1\cdot\log\frac{1\cdot p_1}{p_1\cdot p_1} + p_2\cdot 1\cdot\log\frac{1\cdot p_2}{p_2(p_2+p_3)} \\ &\quad + p_3\cdot 1\cdot\log\frac{1\cdot p_3}{p_3(p_2+p_3)} \\ &= p_1\log\frac{1}{p_1} + (p_2+p_3)\log\frac{1}{p_2+p_3} \\ &= p_1\log\frac{1}{p_1} + (1-p_1)\log\frac{1}{1-p_1} \\ &= h_2(p_1) \end{aligned}$$

where $h_2(\cdot)$ is the binary entropy function. This means that for all p with $p_1 \in [0,1]\backslash\{\frac{1}{2}\}$ we always have

$$I(p,W_*) < 1 = h_2(p_*) = I(p_*,W_*)$$

with $p_* \in \mathcal{P}_1$ arbitrary as defined above.

Next, we define the channel

$$\hat{W} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

and for $\mu \in [0,1]$ we have

$$W_{1,\mu} = (1-\mu)W_* + \mu\hat{W}.$$

Then for $p \in \mathcal{P}(\mathcal{X})$ arbitrary, we always have

$$I(p,W_{1,\mu}) \le (1-\mu)I(p,W_*) + \mu I(p,\hat{W}).$$

We now consider the set

$$\mathcal{P}_2 = \Big\{p=(p_1,p_2,p_3)\in\mathcal{P}(\mathcal{X}) : p_2 = \frac{1}{2} \text{ and } p_1+p_3 = \frac{1}{2}\Big\}.$$

Similarly, we can show for the channel $\hat{W}$ that

$$\max_{p\in\mathcal{P}(\mathcal{X})} I(p,\hat{W}) = 1 = I(\hat{p},\hat{W})$$

with $\hat{p} \in \mathcal{P}_2$ arbitrary. Further, we have

$$\mathcal{P}_1 \cap \mathcal{P}_2 = \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \\ 0 \end{pmatrix}.$$

For $p \in \mathcal{P}(\mathcal{X})$, $p \neq (\frac{1}{2},\frac{1}{2},0)$, we must have

$$I(p,W_*) < 1 \quad \text{or} \quad I(p,\hat{W}) < 1.$$

Thus, for arbitrary $p \in \mathcal{P}(\mathcal{X})$ with $p \in \mathcal{P}(\mathcal{X})$, $p \neq (\frac{1}{2},\frac{1}{2},0)$ we always have

$$\begin{aligned} I(p,W_{1,\mu}) &\le (1-\mu)I(p,W_*) + \mu I(p,\hat{W}) \\ &< (1-\mu) + \mu \\ &= 1. \end{aligned}$$

For

$$p_*^{(1)} = \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \\ 0 \end{pmatrix}$$

we have

$$I(p_*^{(1)},W_{1,\mu}) = 1$$

for $\mu \in [0,1]$. Consequently, for channel $W_{1,\mu}$ for $\mu \in (0,1)$ there is exactly one optimal input distribution, i.e., $\mathcal{P}_{\text{opt}}(W_{1,\mu}) = \{p_*^{(1)}\}$.

Similarly, one can show that for channel $W_{2,\mu}$ for $\mu \in (0,1)$ there is exactly one optimal input distribution, i.e., $\mathcal{P}_{\text{opt}}(W_{2,\mu}) = \{p_*^{(2)}\}$ given by

$$p_*^{(2)} = \begin{pmatrix} \frac{1}{2} \\ 0 \\ \frac{1}{2} \end{pmatrix}.$$

B. Non-Computability of an Optimal Input Distribution

Now we are in the position to prove Theorem 2. We start with the case $|\mathcal{X}| = 3$ and $|\mathcal{Y}| = 2$ and prove the desired result by contradiction. For this purpose, we assume that there exists a function $F \in \mathcal{M}_{\text{opt}}(\mathcal{X};\mathcal{Y})$ that is Banach-Mazur computable. This means that every computable sequence $(W_n)_{n\in\mathbb{N}}$ of computable channels $W_n \in \mathcal{CH}_c(\mathcal{X};\mathcal{Y})$ is mapped to a computable sequence $(p_n)_{n\in\mathbb{N}}$ of computable input distributions $p_n \in \mathcal{P}_c(\mathcal{X})$ for all $n \in \mathbb{N}$. For the set of optimal input distributions (4) we always have $\mathcal{P}_{\text{opt}}(W) \neq \emptyset$. Further, let F be an arbitrary function as in (5) and

$$F(W) \in \mathcal{P}_{\text{opt}}(W),$$

i.e., F maps every channel to an optimal input distribution for this channel.

For our previously defined channel W_*, cf. (8), we therefore have

$$F(W_*) \in \mathcal{P}_{\text{opt}}(W_*) = \mathcal{P}_1.$$

For $\mu \in (0,1)$, we further have

$$F(W_{1,\mu}) = p_*^{(1)}$$

since $\mathcal{P}_{\text{opt}}(W_{1,\mu}) = \{p_*^{(1)}\}$ for $\mu \in (0,1)$.

For $\mu \in (0,1)$ we also have

$$F(W_{2,\mu}) = p_*^{(2)}$$

since $\mathcal{P}_{\text{opt}}(W_{2,\mu}) = \{p_*^{(2)}\}$ for $\mu \in (0,1)$. We have $p_*^{(1)} \in \mathcal{P}_1$, $p_*^{(2)} \in \mathcal{P}_1$, and $\|p_*^{(1)} - p_*^{(2)}\| = 1$. With this, we obtain

$$\begin{aligned} 1 &= \big\|p_*^{(1)} - p_*^{(2)}\big\|_{\ell_1} \\ &= \big\|p_*^{(1)} - F(W_*) + F(W_*) - p_*^{(2)}\big\|_{\ell_1} \\ &\le \big\|p_*^{(1)} - F(W_*)\big\|_{\ell_1} + \big\|F(W_*) - p_*^{(2)}\big\|_{\ell_1} \\ &\le 2\max\Big\{\big\|p_*^{(1)} - F(W_*)\big\|_{\ell_1}, \big\|p_*^{(2)} - F(W_*)\big\|_{\ell_1}\Big\} \end{aligned}$$

so that

$$\max\Big\{\big\|p_*^{(1)} - F(W_*)\big\|_{\ell_1}, \big\|p_*^{(2)} - F(W_*)\big\|_{\ell_1}\Big\} \ge \frac{1}{2}.$$

Let $\mathcal{A} \subset \mathbb{N}$ be a recursively enumerable set that is not recursive, cf. Section II. Let $g : \mathbb{N} \to \mathcal{A}$ be a computable function where for each $m \in \mathcal{A}$ there exists a k with $g(k) = m$ and $g(k_1) \neq g(k_2)$ for $k_1 \neq k_2$.

Let $\mathfrak{T}_{\mathcal{A}}$ be a Turing machine that accepts exactly the set $\mathcal{A}$, i.e., $\mathfrak{T}_{\mathcal{A}}$ stops for input $k \in \mathbb{N}$ if and only if $k \in \mathcal{A}$. Otherwise,

$\mathfrak{T}_{\mathcal{A}}$ runs forever and does not stop. For $k \in \mathbb{N}$ and $n \in \mathbb{N}$, we define the function

$$q(k,n) = \begin{cases} 2^{s+2} & \text{if } \mathfrak{T}_{\mathcal{A}} \text{ stops for input } k \text{ after } s \leq n \text{ steps} \\ 2^{n+2} & \text{if } \mathfrak{T}_{\mathcal{A}} \text{ does not stop for input } k \text{ after } n \text{ steps.} \end{cases}$$

Note that $q : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ is a computable function.

Let $k, n \in \mathbb{N}$ be arbitrary. If k is odd, i.e., $k \in \mathcal{O}$ with $\mathcal{O} \subset \mathbb{N}$ the set of all odd numbers, then we have $k = 2l - 1$, $l \geq 1$, $l \in \mathbb{N}$, and we consider the channel $W_{k,n} \coloneqq W_{1,\frac{1}{q(l,n)}}$. If k is even, i.e., $k \in \mathcal{E}$ with $\mathcal{E} \subset \mathbb{N}$ the set of all even numbers, then we have $k = 2l$, $l \geq 1$, $l \in \mathbb{N}$, and we consider $W_{k,n} \coloneqq W_{2,\frac{1}{q(l,n)}}$. Note that in both cases, l is a function of k. With this, $(W_{k,n})_{k\in\mathbb{N},n\in\mathbb{N}}$ is a computable double sequence.

Now, we define the following sequence $(W_k^*)_{k\in\mathbb{N}}$. We will later show in the proof that $(W_k^*)_{k\in\mathbb{N}}$ is even a computable sequence of computable channels. For $k \in \mathbb{N}$, k is either odd or even:

1) $k \in \mathcal{O}$ odd, i.e., $k = 2l-1$, $l \geq 1$, $l \in \mathbb{N}$. If $l \in \mathcal{A}$, then we set $W_k^* \coloneqq W_{1,\frac{1}{2^{s+2}}}$ with $\mathfrak{T}_{\mathcal{A}}$ has stopped for input l after s steps. If $l \notin \mathcal{A}$, then we set $W_k^* \coloneqq W_*$.
2) $k \in \mathcal{E}$ even, i.e., $k = 2l$, $l \geq 1$, $l \in \mathbb{N}$. If $l \in \mathcal{A}$, then we set $W_k^* \coloneqq W_{2,\frac{1}{2^{s+2}}}$ with $\mathfrak{T}_{\mathcal{A}}$ has stopped for input l after s steps. If $l \notin \mathcal{A}$, then we set $W_k^* \coloneqq W_*$.

Next, we show that the double sequence $(W_{k,n})_{k\in\mathbb{N},n\in\mathbb{N}}$ converges effectively to the sequence $(W_k^*)_{k\in\mathbb{N}}$. This implies that $(W_k^*)_{k\in\mathbb{N}}$ is a computable sequence of computable channels. Further, we show that for all $k \in \mathbb{N}$ and $n \in \mathbb{N}$ we have

$$D(W_k^*, W_{k,n}) < \frac{1}{2^n} \tag{9}$$

so that $(W_{k,n})_{k\in\mathbb{N},n\in\mathbb{N}}$ indeed converges effectively.

Let $k \in \mathbb{N}$ be arbitrary. We first consider the case $k \in \mathcal{O}$, i.e., $k = 2l-1$, $l \geq 1$, $l \in \mathbb{N}$. If $l \notin \mathcal{A}$, we have $W_k^* = W_*$ so that

$$D\big(W_k^*, W_{k,n}\big) = D\big(W_*, W_{1,\frac{1}{2^{n+2}}}\big) = \frac{2}{2^{n+2}} < \frac{1}{2^n}$$

which already shows (9) for this case. In the other case, if $l \in \mathcal{A}$, we have $W_k^* = W_{1,\frac{1}{2^{s+2}}}$, where s is the actual number of steps after which the Turing machine $\mathfrak{T}_{\mathcal{A}}$ stopped for input l. Now, let $n \in \mathbb{N}$ be arbitrary. For $n \geq s$ we have

$$W_{k,n} = W_{2l-1,n} = W_{1,\frac{1}{2^{s+2}}} = W_k^*$$

so that

$$D(W_k^*, W_{k,n}) = 0.$$

For $n < s$ we have $W_{k,n} = W_{1,\frac{1}{2^{n+2}}}$ so that

$$\begin{aligned} D(W_k^*, W_{k,n}) &= D(W_{1,\frac{1}{2^{s+2}}}, W_{1,\frac{1}{2^{n+2}}}) \\ &= \Big|\Big(1 - \frac{1}{2^{s+2}}\Big) - \Big(1 - \frac{1}{2^{n+2}}\Big)\Big| + \Big|\frac{1}{2^{s+2}} - \frac{1}{2^{n+2}}\Big| \\ &= 2\Big|\frac{1}{2^{n+2}} - \frac{1}{2^{s+2}}\Big| < 2\frac{1}{2^{n+2}} < \frac{1}{2^n} \end{aligned}$$

which shows (9) for this case as well.

The proof for even numbers $k \in \mathcal{E}$ follows as above for odd numbers $k \in \mathcal{O}$ and is omitted for brevity. As a consequence, $(W_k^*)_{k\in\mathbb{N}}$ is a computable sequence of computable channels. If the function F is Banach-Mazur computable, then the sequence $(F(W_k^*))_{k\in\mathbb{N}}$ must be a computable sequence of computable input distributions in $\mathcal{P}_c(\mathcal{X})$.

We consider the computable sequence

$$\big(F(W_k^*) - F(W_*)\big)_{k\in\mathbb{N}} \tag{10}$$

and the following Turing machine: For $l \in \mathbb{N}$ we start two Turing machines in parallel.

The first Turing machine $\mathfrak{T}_1$ is given by $\mathfrak{T}_1 = \mathfrak{T}_{\mathcal{A}}$, i.e., for input l it runs the algorithm for $\mathfrak{T}_{\mathcal{A}}$ step by step.

The second Turing machine is given as follows. We compute $n = 2l-1$ and also $F(W_{2l-1}^*) - F(W_*)$ which is possible since (10) is a computable sequence. We compute $\|F(W_{2l-1}^*) - F(W_*)\|_{\ell_1}$. In parallel, we further compute $n = 2l$ and also $F(W_{2l}^*) - F(W_*)$ and $\|F(W_{2l}^*) - F(W_*)\|_{\ell_1}$. We now compute

$$r_l = \max\big\{\|F(W_{2l-1}^*) - F(W_*)\|_{\ell_1}, \|F(W_{2l}^*) - F(W_*)\|_{\ell_1}\big\}.$$

We now use the Turing machine $\mathfrak{T}_{<\frac{1}{4}}$ from [27, page 14] and test if $r_l < \frac{1}{4}$ is true. Our second Turing machine $\mathfrak{T}_2$ stops if and only if the Turing machine $\mathfrak{T}_{<\frac{1}{4}}$ stops for input r_l.

We start both Turing machines in parallel in such a way that the computing steps are synchronous. Whenever the first Turing machine stops, we set $l \in \mathcal{A}$. Otherwise, if the second Turing machine stops, we set $l \notin \mathcal{A}$. The first Turing machine stops if and only if $l \in \mathcal{A}$. The second Turing machine stops if and only if $r_l < \frac{1}{4}$. As for $l \in \mathcal{A}$ we have $r_l \geq \frac{1}{2}$ and for $l \notin \mathcal{A}$ we have $r_l = 0$, the second Turing machine stops if and only if $l \notin \mathcal{A}$.

With this, we have obtained a Turing machine $\mathfrak{T}_*$ that always decides for $l \in \mathbb{N}$ whether $l \in \mathcal{A}$ or $l \notin \mathcal{A}$. This means that $\mathcal{A}$ must be a recursive set which is a contradiction to our initial assumption. Thus, the function F is not Banach-Mazur computable which proves the desired result for the case $|\mathcal{X}| = 3$ and $|\mathcal{Y}| = 2$.

Finally, we outline how the proof extends to arbitrary $|\mathcal{X}| \geq 3$ and $|\mathcal{Y}| \geq 2$. In this case, for the set $\mathcal{CH}_c(\mathcal{X};\mathcal{Y})$ we consider the subset $\underline{\mathcal{CH}}_c(\mathcal{X};\mathcal{Y})$ of all channels $W \in \mathcal{CH}_c(\mathcal{X};\mathcal{Y})$ and choose an arbitrary channel $\underline{W} \in \mathcal{CH}_c(\mathcal{X}_1;\mathcal{Y}_1)$ with $|\mathcal{X}_1| = 3$ and $|\mathcal{Y}_1| = 2$. We set

$$W(y|x) = \begin{cases} \underline{W}(y|x) & y \in \{1,2\} x \in \{1,2,3\} \\ 0 & y \in \{3,...,|\mathcal{Y}|\} x \in \{1,2,3\} \end{cases} \tag{11}$$

as well as

$$W(\cdot|x) = \underline{W}(\cdot|1) \quad x \in \{4,...,|\mathcal{X}|\}. \tag{12}$$

As above, we assume that $F \in \mathcal{M}_{\text{opt}}(\mathcal{X};\mathcal{Y})$ is a Banach-Mazur computable function that computes an optimal input distribution for the set $\mathcal{CH}_c(\mathcal{X};\mathcal{Y})$. Then we always have $F(W) \in \mathcal{P}(\mathcal{X})$ for $W \in \mathcal{CH}_c(\mathcal{X};\mathcal{Y})$. For $\underline{W} \in \mathcal{CH}_c(\mathcal{X}_1;\mathcal{Y}_1)$ we can immediately compute an optimal input distribution $p_1^* \in \mathcal{P}_{\text{opt}}(\underline{W})$ as follows. We take W which is constructed as above in (11)-(12). Let $W \in \mathcal{CH}_c(\mathcal{X};\mathcal{Y})$ and consider $p(W) \coloneqq F(W)$. With

$$p(W) = \begin{pmatrix} p_1(W) \\ \vdots \\ p_{|\mathcal{X}|}(W) \end{pmatrix}$$

we set

$$p_1^*(\underline{W}) \coloneqq p_1(W) + \sum_{x=4}^{|\mathcal{X}|} p_x(W) \tag{13}$$

and

$$p_2^*(\underline{W}) \coloneqq p_2(W), \tag{14}$$

$$p_3^*(\underline{W}) \coloneqq p_3(W). \tag{15}$$

For $\underline{W} \in \mathcal{CH}(\mathcal{X}_1;\mathcal{Y}_1)$ we consider the mapping

$$G(\underline{W}) = \begin{pmatrix} p_1^*(\underline{W}) \\ p_2^*(\underline{W}) \\ p_3^*(\underline{W}) \end{pmatrix}$$

which is defined by (13)-(15). The mapping G is a composition of the following components: 1) it constructs from $\underline{W}$ the channel W according to (11)-(12); 2) it applies the function F on W; and 3) it applies the operations (13)-(15) on F. The construction (11)-(12) and also the operations (13)-(15) are Borel-Turing computable. Since F is further Banach-Mazur computable by assumption, the mapping G must be Banach-Mazur computable as well. However, we have $p_* \in \mathcal{P}_{\text{opt}}(W)$. This is a contradiction since for $|\mathcal{X}_1| = 3$ and $|\mathcal{Y}_1| = 2$ all functions $G \in \mathcal{M}_{\text{opt}}(\mathcal{X}_1;\mathcal{Y}_1)$ can not be Banach-Mazur computable. This proves the general case and therewith completes the proof of Theorem 2. ■

V. Non-Approximability of the Optimizer

In this section we present the proof of Theorem 3. We prove the result by contradiction. Therefore, we assume that there exists a function $F \in \mathcal{M}_{\text{opt}}(\mathcal{X};\mathcal{Y})$ such that there is a function F_1 with

$$\sup_{W \in \mathcal{CH}_c(\mathcal{X};\mathcal{Y})} \left|F(W) - F_1(W)\right| = \beta < 1$$

that is Banach-Mazur computable. Then, there exists a computable real number α with $\beta \leq \alpha < 1$.

We now consider the computable sequence $(W_n^*)_{n\in\mathbb{N}}$ as used in the proof of Theorem 2. For $l \in \mathbb{N}$, let

$$\left\|F_1(W_{2l}^*) - F(W_{2l}^*)\right\|_{\ell_1} \leq \alpha$$

and

$$\left\|F_1(W_{2l-1}^*) - F(W_{2l-1}^*)\right\|_{\ell_1} \leq \alpha$$

be satisfied. Then, we also have for $l \in \mathcal{A}$ the following:

$$\begin{aligned}
1 &= \left\|F(W_{2l}^*) - F(W_{2l-1}^*)\right\|_{\ell_1} \\
&= \left\|F(W_{2l}^*) - F_1(W_{2l}^*) + F_1(W_{2l}^*) - F_1(W_{2l-1}^*) \right. \\
&\qquad \left. + F_1(W_{2l-1}^*) - F(W_{2l-1}^*)\right\|_{\ell_1} \\
&\leq \left\|F(W_{2l}^*) - F_1(W_{2l}^*)\right\|_{\ell_1} + \left\|F_1(W_{2l}^*) - F_1(W_{2l-1}^*)\right\|_{\ell_1} \\
&\qquad + \left\|F_1(W_{2l-1}^*) - F(W_{2l-1}^*)\right\|_{\ell_1} \\
&\leq 2\alpha + \left\|F_1(W_{2l}^*) - F_1(W_{2l-1}^*)\right\|_{\ell_1}.
\end{aligned}$$

Therefore, it holds that

$$\left\|F_1(W_{2l}^*) - F_1(W_{2l-1}^*)\right\|_{\ell_1} \geq 1 - 2\alpha = c_1 > 0$$

which implies that

$$\begin{aligned}
c_1 &= \left\|F_1(W_{2l}^*) - F_1(W_*) + F_1(W_*) - F_1(W_{2l-1}^*)\right\|_{\ell_1} \\
&\leq \left\|F_1(W_{2l}^*) - F_1(W_*)\right\|_{\ell_1} + \left\|F_1(W_*) - F_1(W_{2l-1}^*)\right\|_{\ell_1} \\
&\leq 2\max\left\{\left\|F_1(W_{2l}^*) - F_1(W_*)\right\|_{\ell_1}, \right. \\
&\qquad \left. \left\|F_1(W_*) - F_1(W_{2l-1}^*)\right\|_{\ell_1}\right\} \\
&=: 2r_l^*.
\end{aligned}$$

We conclude that

$$r_l^* \geq \frac{c_1}{2} > 0. \tag{16}$$

For $l \in \mathbb{N}$ and $l \notin \mathcal{A}$,

$$F_1(W_{2l}^*) = F_1(W_*)$$

and

$$F_1(W_{2l-1}^*) = F_1(W_*)$$

are satisfied. Accordingly, we can use the same Turing machine $\mathfrak{T}_{<\frac{1}{4}}$ as in the proof of Theorem 2 for the input r_l^* in (16). The Turing machine $\mathfrak{T}_{<\frac{1}{4}}(r_l^*)$ stops if and only if $l \notin \mathcal{A}$. Thus, we can construct a Turing machine as in the proof of Theorem 2 that decides for every $l \in \mathbb{N}$ whether $l \in \mathcal{A}$ or $l \notin \mathcal{A}$. This is, again, a contradiction to the initial assumption completing the proof of Theorem 3. ■

VI. Conclusion

The channel capacity describes the maximum rate at which a source can be reliably transmitted. Capacity expressions are usually given by entropic quantities that are optimized over all possible input distributions. Evaluating such capacity expressions and finding corresponding optimal input distributions that maximize these capacity expressions is a common and important task in information and communication theory. Several algorithms including the Blahut-Arimoto algorithm have been proposed to algorithmically compute these quantities. In this work, we have shown that there exists no algorithm or Turing machine that takes a DMC as input and then computes an input distribution that maximizes the capacity. Although capacity-achieving input distributions have been found analytically for some specific DMCs, this does not immediately mean that capacity-achieving input distributions can be algorithmically computed by a Turing that takes a DMC of interest as input. We have further shown that it is not even possible to algorithmically approximate this distribution. These results have implications for the Blahut-Arimoto algorithm. In particular, as we have noted, there is no stopping criterion for the computation of the input distribution, and our results imply that such a computable stopping criterion cannot exist, providing a negative answer to the open question of whether one does.

Appendix

A. Example of a Non-Computable Function

Here, we show that for $x \in [0,1] \cap \mathbb{R}_c$ the function

$$f(t) = e^{-x}$$

is not exactly computable on Turing machines, but only approximable.

By the remainder theorem of Lagrange, we get for $x \in [0,1]$:

$$f(x) = \sum_{l=0}^{n} \frac{(-1)^l}{l!} x^l + \frac{1}{(n+1)!} f^{(n+1)}(\vartheta_x) x^{n+1}$$

with $\vartheta_x \in [0,x]$ a suitable number. With this, we get

$$\left| f(x) - \sum_{l=1}^{n} \frac{(-1)^l}{l!} x^l \right| = \frac{1}{(n+1)!} e^{-\vartheta_x} x^{n+1} \leq \frac{1}{(n+1)!}$$

and

$$(n+1)! > 2^n, \qquad n \geq 2.$$

Assume now that we have a sequence $(r_n)_{n\in\mathbb{N}}$ of rational numbers with

$$|x - r_n| < \frac{1}{2^n}$$

so that

$$\begin{aligned}
&\left| f(x) - \sum_{l=0}^{n} \frac{(-1)^l}{l!} (r_n)^l \right| \\
&\quad = \left| f(x) - f(r_n) + f(r_n) - \sum_{l=0}^{n} \frac{(-1)^l}{l!} (r_n)^l \right| \\
&\quad \leq |f(x) - f(r_n)| + \left| f(r_n) - \sum_{l=0}^{n} \frac{(-1)^l}{l!} (r_n)^l \right| \\
&\quad < |f(x) - f(r_n)| + \frac{1}{2^n}, \qquad n \geq 2.
\end{aligned}$$

Now, the mean value theorem implies that

$$|f(x) - f(r_n)| = |f'(\xi_{x,n})| \cdot |x - r_n|$$

with $\xi_{x,n} \in [x - r_n, x + r_n]$ being a suitable number. This yields

$$|f(x) - f(r_n)| \leq |x - r_n| < \frac{1}{2^n}.$$

With $y_n := \sum_{l=0}^{n} \frac{(-1)^l}{l!} (r_n)^l$, we obtain

$$|f(x) - y_n| < \frac{1}{2^n} + \frac{1}{2^n} = \frac{1}{2^{n-1}},$$

i.e., the algorithm

$$(r_n)_{n\in\mathbb{N}} \to (y_n)_{n\in\mathbb{N}}$$

maps a representation of x into a representation of $f(x)$. This algorithm converges effectively.

From this calculation we immediately see how the function f can be approximated. Whenever f needs to be approximated in such a way that the error satisfies $\frac{1}{2^n}$, we use the polynomial as given above and compute it accordingly. For this, it is important to find suitable sequences of polynomials. Note that the polynomials in this sequence needs to be computable as well as the sequence itself needs to be a computable sequence, since otherwise, we are not able to evaluate the approximation process algorithmically. Note that this does not mean that every sequence of approximations of f is also a suitable sequence for our purpose.

B. Binary Entropy and Transcendental Numbers

For the following, we need Hilbert's Seventh Problem which is restated next for completeness.

> **Hilbert's Seventh Problem.** Let $a \notin \{0,1\}$ be an algebraic number (i.e., a root of a non-zero polynomial with integer coefficients) and let b be an irrational and algebraic number. Is a^b always a transcendental number (i.e., not algebraic)?

Remark 10. A positive answer to this question was then first given in 1934 by Gelfond [36] and subsequently refined in 1935 by Schneider [37]. Later this was generalized by Baker for which he was awarded a Fields Medal in 1970, cf. [38].

We further need the following observation.

Lemma 1. Let $n \in \mathbb{N}$ and $t \in \mathbb{N}$ be arbitrary. Then n and n^t are divisible by the same prime numbers.

Proof: Let $n = \prod_{l=1}^{r} p_l$ be the unique prime factorization of n. Note that in factorization, certain prime factors may appear multiple times. Then, $n^t = \prod_{l=1}^{r} (p_l)^t$ is a prime factorization of n^t. As this factorization is unique, both n and n^t must have the same prime factors. ∎

We now prove the following result.

Theorem 4. Let $p \in \mathbb{Q}$ with $p \notin \{0, \frac{1}{2}, 1\}$. Then, $h_2(p)$ is a transcendental number.

Proof: Let

$$h_2(p) = p \log \frac{1}{p} + (1-p) \log \frac{1}{1-p}$$

be the binary entropy, which can be equivalently be expressed as

$$2^{h_2(p)} = \left(\frac{1}{p}\right)^p \left(\frac{1}{1-p}\right)^{1-p}. \tag{17}$$

Let $p \in \mathbb{Q}$ with $p \in (0,1)$, $p \notin \{0, \frac{1}{2}, 1\}$ be arbitrary. Then, we can express p as $p = \frac{n}{m}$, $n < m$, $n, m \in \mathbb{N}$, and assume without loss of generality that n and m are coprime. With this, we can write

$$\left(\frac{1}{p}\right)^p = \left(\frac{m}{n}\right)^{\frac{n}{m}}$$

and conclude that the number $(\frac{1}{p})^p$ is a root of the polynomial $x^m - (\frac{m}{n})^n$ and therewith also of the polynomial $n^n x^m - m^n$. Thus, $(\frac{1}{p})^p$ is an algebraic number. Similarly, one can show that $(\frac{1}{1-p})^{1-p}$ is an algebraic number so that $2^{h_2(p)}$ as in (17) is also an algebraic number.

Now, we can use the Gelfond-Schneider theorem, i.e., the solution to Hilbert's Seventh Problem, cf. for example [38]. As $2^{h_2(p)}$ is an algebraic number, $h_2(p)$ must be either rational or transcendental. Since otherwise, if $h_2(p)$ would be algebraic and irrational, then $2^{h_2(p)}$ would be transcendental.

Next, we want to show by contradiction that $h_2(p)$ cannot be rational. Since $p \in (0,1)$, $p \neq \frac{1}{2}$, we have $h_2(p) \in (0,1)$. For this purpose, we assume that $h_2(p)$ is rational so that it can be expressed as $h_2(p) = \frac{u}{v}$ with $0 < u < v$, $u, v \in \mathbb{N}$,

and u, v coprime without loss of generality. We further must have $v > 1$. This would imply that

$$2^{\frac{u}{v}} = \left(\frac{m}{n}\right)^{\frac{n}{m}} \left(\frac{1}{1-\frac{n}{m}}\right)^{1-\frac{n}{m}}$$
$$= \left(\frac{m}{n}\right)^{\frac{n}{m}} \left(\frac{m}{m-n}\right)^{\frac{m-n}{m}}$$

so that

$$2^{mu} = \left(\frac{m}{n}\right)^{nv} \left(\frac{m}{m-n}\right)^{(m-n)v}$$

or equivalently

$$2^{mu}(n)^{nv}(m-n)^{(m-n)v} = (m)^{mv}.$$

Note that $m-n \geq 1$ and further $nv \in \mathbb{N}$, $nv > 1$, since $v > 1$.

If $n = 1$, then

$$2^{mu}(m-1)^{(m-1)v} = (m)^{mv}.$$

Lemma 1 and the uniqueness of the prime factorization would then imply that every prime factor of $m = 1$ must be a prime factor m as well. However, this is not possible.

If $n > 1$, then Lemma 1 implies that every prime factor of n must also be a prime factor of m. However, this is not possible, since n and m are coprime. As a consequence, $h_2(p)$ cannot be a rational number. Finally, we conclude that $h_2(p)$ must be a transcendental number which completes the proof. ■

REFERENCES

[1] S. Arimoto, "An algorithm for computing the capacity of arbitrary discrete memoryless channels," *IEEE Trans. Inf. Theory*, vol. 18, no. 1, pp. 14–20, Jan. 1972.

[2] R. E. Blahut, "Computation of channel capacity and rate-distortion functions," *IEEE Trans. Inf. Theory*, vol. 18, no. 4, pp. 460–473, Jul. 1972.

[3] I. Csiszár, "On the computation of rate-distortion functions," *IEEE Trans. Inf. Theory*, vol. 20, no. 1, pp. 122–124, Jan. 1974.

[4] I. Csiszár and G. Tusnády, "Information geometry and alternating minimization procedures," *Statistics and Decisions, Supplement Issue 1*, pp. 205–237, 1984.

[5] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2nd ed. Wiley & Sons, 2006.

[6] R. W. Yeung, *Information Theory and Network Coding*. Springer-Verlag, 2008.

[7] F. Dupuis, W. Yu, and F. M. J. Willems, "Blahut-Arimoto algorithms for computing channel capacity and rate-distortion with side information," in *Proc. IEEE Int. Symp. Inf. Theory*, Chicago, IL, USA, Jun. 2004, p. 179.

[8] P. O. Vontobel, A. Kavcic, D. M. Arnold, and H.-A. Loeliger, "A generalization of the Blahut–Arimoto algorithm to finite-state channels," *IEEE Trans. Inf. Theory*, vol. 54, no. 5, pp. 1887–1918, May 2008.

[9] T. J. Oechtering, M. Andersson, and M. Skoglund, "Arimoto-Blahut algorithm for the bidirectional broadcast channel with side information," in *Proc. IEEE Inf. Theory Workshop*, Taormina, Italy, Oct. 2009, p. 394–398.

[10] I. Naiss and H. H. Permuter, "Extension of the Blahut–Arimoto algorithm for maximizing directed information," *IEEE Trans. Inf. Theory*, vol. 59, no. 1, pp. 204–222, Jan. 2013.

[11] K. F. Trillingsgaard, O. Simeone, P. Popovski, and T. Larsen, "Blahut-Arimoto algorithm and code design for action-dependent source coding problems," in *Proc. IEEE Int. Symp. Inf. Theory*, Istanbul, Turkey, Jul. 2013, pp. 1192–1196.

[12] Y. Ugur, I. E. Aguerri, and A. Zaidi, "A generalization of Blahut-Arimoto algorithm to compute rate-distortion regions of multiterminal source coding under logarithmic loss," in *Proc. IEEE Inf. Theory Workshop*, Kaohsiung, Taiwan, Nov. 2017, pp. 349–353.

[13] H. Li and N. Cai, "A Blahut-Arimoto type algorithm for computing classical-quantum channel capacity," in *Proc. IEEE Int. Symp. Inf. Theory*, Paris, France, Jul. 2019, pp. 255–259.

[14] N. Ramakrishnan, R. Iten, V. Scholz, and M. Berta, "Quantum Blahut-Arimoto algorithms," in *Proc. IEEE Int. Symp. Inf. Theory*, Los Angeles, CA, USA, Jun. 2020, pp. 1909–1914.

[15] A. Ephremides and B. Hajek, "Information theory and communication networks: An unconsummated union," *IEEE Trans. Inf. Theory*, vol. 44, no. 6, pp. 2416–2434, Oct. 1998.

[16] A. M. Turing, "On computable numbers, with an application to the Entscheidungsproblem," *Proc. London Math. Soc.*, vol. 2, no. 42, pp. 230–265, 1936.

[17] ——, "On computable numbers, with an application to the Entscheidungsproblem. A correction," *Proc. London Math. Soc.*, vol. 2, no. 43, pp. 544–546, 1937.

[18] K. Weihrauch, *Computable Analysis - An Introduction*. Berlin, Heidelberg: Springer-Verlag, 2000.

[19] J. Avigad and V. Brattka, "Computability and analysis: The legacy of Alan Turing," in *Turing's Legacy: Developments from Turing's Ideas in Logic*, R. Downey, Ed. Cambridge, UK: Cambridge University Press, 2014.

[20] K. Gödel, "Die Vollständigkeit der Axiome des logischen Funktionenkalküls," *Monatshefte für Mathematik*, vol. 37, no. 1, pp. 349–360, 1930.

[21] ——, "On undecidable propositions of formal mathematical systems," *Notes by Stephen C. Kleene and Barkely Rosser on Lectures at the Institute for Advanced Study, Princeton, NJ*, 1934.

[22] S. C. Kleene, *Introduction to Metamathematics*. Van Nostrand, New York: Wolters-Noordhoffv, 1952.

[23] M. Minsky, "Recursive unsolvability of Post's problem of 'tag' and other topics in theory of Turing machines," *Ann. Math.*, vol. 74, no. 3, pp. 437–455, 1961.

[24] H. Boche, R. F. Schaefer, and H. V. Poor, "Secure communication and identification systems – Effective performance evaluation on Turing machines," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 1013–1025, 2020.

[25] ——, "Shannon meets Turing: Non-computability of the finite state channel capacity," *Commun. Inf. Syst.*, vol. 20, no. 2, pp. 81–116, 2020, invited in honor of Prof. Thomas Kailath on the occasion of his 85th birthday.

[26] ——, "Coding for non-iid sources and channels: Entropic approximations and a question of Ahlswede," in *Proc. IEEE Inf. Theory Workshop*, Visby, Sweden, Aug. 2019, pp. 1–5.

[27] M. B. Pour-El and J. I. Richards, *Computability in Analysis and Physics*. Cambridge: Cambridge University Press, 2017.

[28] H. Boche, R. F. Schaefer, S. Baur, and H. V. Poor, "On the algorithmic computability of the secret key and authentication capacity under channel, storage, and privacy leakage constraints," *IEEE Trans. Signal Process.*, vol. 67, no. 17, pp. 4636–4648, Sep. 2019.

[29] R. I. Soare, *Recursively Enumerable Sets and Degrees*. Berlin, Heidelberg: Springer-Verlag, 1987.

[30] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, Jul. 1948.

[31] A. El Gamal and Y.-H. Kim, *Network Information Theory*. Cambridge, UK: Cambridge University Press, 2011.

[32] R. G. Bartle and D. R. Sherbert, *Introduction to Real Analysis*, 3rd ed. New York: Wiley, 2000.

[33] T. Y. Chow, "What is a closed-form number?" *Amer. Math. Monthly*, vol. 106, pp. 440–448, 1999.

[34] J. M. Borwein and R. E. Crandall, "Closed forms: What they are and why we care," *Notices of the American Mathematical Society*, vol. 60, pp. 50–65, 2013.

[35] I. N. Herstein, *Topics in Algebra*, 2nd ed. Wiley, 1975.

[36] A. Gelfond, "On Hilbert's seventh problem," *Doklady Akademii Nauk SSSR*, p. 177–182, 1934.

[37] T. Schneider, "Transzendenzuntersuchungen periodischer Funktionen I. Transzendenz von Potenzen," *Journal für die reine und angewandte Mathematik*, no. 172, pp. 65–69, 1935.

[38] A. Baker, *Transcendental Number Theory*. Cambridge, UK: Cambridge University Press, 1975.