

Shannon Meets Turing: Non-Computability and Non-Approximability of the Finite State Channel Capacity

In honor of Prof. Thomas Kailath on the occasion of his 85th birthday

HOLGER BOCHE, RAFAEL F. SCHAEFER, AND H. VINCENT POOR

The capacity of finite state channels (FSCs) has been established as the limit of a sequence of multi-letter expressions only and, despite tremendous effort, a corresponding finite-letter characterization remains unknown to date. This paper analyzes the capacity of FSCs from a fundamental, algorithmic point of view by studying whether or not the corresponding achievability and converse bounds on the capacity can be computed algorithmically. For this purpose, the concept of Turing machines is used which provide the fundamental performance limits of digital computers. To this end, computable continuous functions are studied and properties of computable sequences of such functions are identified. It is shown that the capacity of FSCs is not Banach-Mazur computable which is the weakest form of computability. This implies that there is no algorithm (or Turing machine) that can compute the capacity of a given FSC. As a consequence, it is then shown that either the achievability or converse must yield a bound that is not Banach-Mazur computable. This also means that there exist FSCs for which computable lower and upper bounds can never be tight. To this end, it is further shown that the capacity of FSCs is not approximable, which is an even stricter requirement than non-computability. This implies that it is impossible to find a finite-letter entropic characterization of the capacity of general FSCs. All results hold even for finite input and output alphabets and finite state set. Finally, connections to the theory of effective analysis are discussed. Here, results are only allowed to be proved in a constructive way, while existence results, e.g., proved based on the axiom of choice, are forbidden.

1. Introduction

Finite state channels (FSCs) model discrete channels with memory where the channel output depends not only on the current channel input but also on the underlying channel state. The channel state allows the channel output to implicitly depend on previous channel inputs and outputs. FSCs are of significant interest as they allow to model certain types of channel variations appearing in wireless communications including e.g. flat fading and inter-symbol interference [1]. FSCs are relatively simple channels and are usually used for approximations of more complex, time-continuous channels. The theory of time-continuous channels goes back to Kailath’s seminal work [2]. Subsequently, communication over such time-continuous channels has been studied, for example, in [3–6]. But FSCs are also used in molecular communication [7]. In the latter context, the trapdoor channel has been introduced as a simple two-state channel and is studied in [8–11]. This channel is also known as “chemical channel” due to Cover. The *indecomposable finite state channel* (IFSC) is introduced in [12]. Estimating the capacity of flat fading IFSCs is considered in [13]. The compound capacity of FSCs is studied in [14].

Determining the capacity of FSCs is extremely challenging. For example, already for the trapdoor channel, the capacity is unknown. Only a lower bound [10] and an upper bound given by the feedback capacity [11] are known. Recently, a reinforcement learning approach has been presented in [15] to compute the feedback capacity. For general FSCs, a finite-letter characterization of the capacity in closed form is not known to date; only a general formula based on the inf-information rate has been established in [16]. In this paper, we are interested in the existence of “*simple*” capacity expressions and whether or not such capacity expressions for FSCs are algorithmically computable. Both questions are related to each other. For example, a simple capacity expression could be given a single-letter formula with entropic quantities. But it could also be a capacity function which is computable in some sense. The requirement of certain performance functions to be computable is usually implicitly assumed in information theory. Particularly, capacity expressions with entropic quantities in dependence on the communication parameters are usually assumed to be algorithmically computable.

For the question of algorithmic computability, we use the concept of a *Turing machine* [17–19], which 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. Accordingly they provide fundamental performance limits for today's digital computers. Turing machines account for all those problems and tasks that are algorithmically solvable on a classical (i.e., non-quantum) machine. They are further equivalent to the von Neumann-architecture without hardware limitations and the theory of recursive functions, cf. [20–24].

Of particular interest in this work are *computable continuous functions* [25] since such functions can be effectively approximated by computable polynomial sequences which is a very strong requirement on the computability. There are other forms of computability including *Banach-Mazur computability*, which is the weakest form of computability. To this end, Section 2 introduces the computability framework and studies further properties and insights of computable sequences of computable continuous functions and of Banach-Mazur computable functions.

Subsequently, this paper studies FSCs which are properly introduced in Section 3. The general question is addressed of whether or not a finite-letter characterization of the capacity exists at all and whether or not the capacity of FSCs is algorithmically computable. In Section 4 it is shown and argued that either the achievability or converse (or both) must result in a non-computable lower or upper bound, respectively. This bound is not even Banach-Mazur computable (and therewith also not Turing computable) and, as a consequence, the capacity is not Banach-Mazur computable as well. This also means that there exist FSCs for which computable lower and upper bounds can never be tight. Furthermore, it is shown that the capacity of FSCs is not even approximable by computable sequences of computable functions, i.e., it is impossible to approximate the capacity for certain tolerated approximation errors. Note that non-approximability is strictly stronger than non-computability. All these results hold for $|\mathcal{X}| \geq 2$, $|\mathcal{Y}| \geq 2$, and $|\mathcal{S}| \geq 2$ and, thus, we consider the general case without restrictions on the cardinalities of the alphabets. This provides a complete picture, since for $|\mathcal{S}| = 1$ the capacity becomes Turing computable and is given by Shannon's single-letter formula. A similar observation with respect to the Turing computability of the capacity of FSCs has been made in [26], where it has been shown that the capacity of FSCs is in general not Turing computable if the input and state alphabets \mathcal{X} and \mathcal{S} satisfy $|\mathcal{X}| \geq 10$ and $|\mathcal{S}| \geq 62$. This result has been used in [27] to show that for a certain class of entropic formulas, the capacity of time invariant Markov channels cannot be expressed by a finite multi-letter formula. Since this uses [26] as a “black

box input”, it further only holds for $|\mathcal{X}| \geq 10$ and $|\mathcal{S}| \geq 62$. Our proof relies on completely different techniques than those in [26] and [27] which further allows us to show that the capacity of FSCs cannot be characterized by a finite-letter entropic expression for input, output, and state alphabets that satisfy $|\mathcal{X}| \geq 2$, $|\mathcal{Y}| \geq 2$, and $|\mathcal{S}| \geq 2$. We emphasize that these results hold even for all FSCs with finite input and output alphabets and finite state sets. When the state set is allowed to be countably infinite, the capacity of a computable channel need not be a computable real number anymore.¹

2. Computability Framework

Here, we introduce the computability framework based on Turing machines which provides the needed background. Subsequently, we establish some results on computable sequences which are needed afterwards.

2.1. Computable Real Numbers and Functions

The concept of computability and computable real numbers was first introduced by Turing in [17] and [18]. Computable numbers are real numbers that are computable by Turing machines. Since the set of all Turing machines is a countable set, the set of computable real numbers is countable as well. See also the introductory textbook [19] for further details.

A sequence of rational numbers $\{r_n\}_{n \in \mathbb{N}}$ is called a *computable sequence* if there exist recursive functions $a, b, s : \mathbb{N} \rightarrow \mathbb{N}$ with $b(n) \neq 0$ for all $n \in \mathbb{N}$ and

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

cf. [28, 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 $\{r_n\}_{n \in \mathbb{N}}$ such that

$$(2) \quad |x - r_n| < 2^{-n}$$

¹*Notation:* \mathbb{N} , \mathbb{Q} , \mathbb{R} , and \mathbb{R}_c are the sets of non-negative integers, rational numbers, real numbers, and computable real numbers; $\mathcal{P}(\mathcal{X})$ and $\mathcal{P}(\mathcal{Y}|\mathcal{X})$ denote the sets of (conditional) probability distributions on \mathcal{Y} (given \mathcal{X}); $H_2(\cdot)$ is the binary entropy function.

for all $n \in \mathbb{N}$. This means that the computable real number x is completely characterized by the recursive functions $a, b, s : \mathbb{N} \rightarrow \mathbb{N}$. It has the representation (a, b, s) which we also write as $x \sim (a, b, s)$. It is clear that this representation must not be unique and that there might be other recursive functions $a', b', s' : \mathbb{N} \rightarrow \mathbb{N}$ which characterize x , i.e., $x \sim (a', b', s')$.

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 every $x \in \mathcal{X}$. The set of all computable conditional probability distributions $\mathcal{P}_c(\mathcal{Y}|\mathcal{X})$ is defined accordingly, i.e., for $P_{Y|X} : \mathcal{X} \rightarrow \mathcal{P}(\mathcal{Y})$ we have $P_{Y|X}(\cdot|x) \in \mathcal{P}_c(\mathcal{Y})$ for every $x \in \mathcal{X}$. This is important since a Turing machine can only operate on computable real numbers.

Definition 1. A function $f : \mathbb{R}_c \rightarrow \mathbb{R}_c$ is called *Borel computable* if there is an algorithm (or Turing machine) that transforms each given representation (a, b, s) of a computable real number x into a corresponding representation for the computable real number $f(x)$.

Remark 2. From a practical point of view, this can be seen as a minimal requirement for the algorithmic computation of the capacity of a communication system. For this task, an algorithm is needed that takes the communication parameters as inputs to compute the capacity value with a certain precision (e.g. decimal points). In information theory, even for simple problems and questions it cannot be expected that a performance quantity can be exactly numerically computed. For example, for an alphabet \mathcal{X} of dimension $|\mathcal{X}| = 2$, the entropy $H_2(p)$ of an arbitrary rational probability distribution $p \in \mathcal{P}(\mathcal{X})$ with $p \neq (\frac{1}{2}, \frac{1}{2})$ is a transcendental number.

Note that Turing's definition of computability conforms to the definition of Borel computability above. In this paper, we will first consider the notion of a *computable continuous function*, cf. for example [25, Def. A]. For this, let \mathbb{I}_c denote a computable interval, i.e., $\mathbb{I}_c = [a, b]$ with $a, b \in \mathbb{R}_c$.

Definition 3 ([25]). Let $\mathbb{I}_c \subset \mathbb{R}_c$ be a computable interval. A function $f : \mathbb{I}_c \rightarrow \mathbb{R}$ is called *computable continuous* if:

- 1) f is *sequentially computable*, i.e., f maps every computable sequence $\{x_n\}_{n \in \mathbb{N}}$ of points $x_n \in \mathbb{I}_c$ into a computable sequence $\{f(x_n)\}_{n \in \mathbb{N}}$ of real numbers,

- 2) f is *effectively uniformly continuous*, i.e., there is a recursive function $d : \mathbb{N} \rightarrow \mathbb{N}$ such that for all $x, y \in \mathbb{I}_c$ and all $N \in \mathbb{N}$ with

$$\|x - y\| \leq \frac{1}{d(N)}$$

it holds that

$$|f(x) - f(y)| \leq \frac{1}{2^N}.$$

Computable continuous functions are functions which can be effectively approximated by computable sequence of polynomials $\{P_n\}_{n \in \mathbb{N}}$. Here, every polynomial P_n itself is computable, i.e., its order and coefficients can algorithmically be computed, cf. [25]. Note that the coefficients of these polynomials are usually rational numbers.

There are other forms of computability including *Banach-Mazur computability*, which is the weakest form of computability. In particular, Borel computability and computable continuous functions imply Banach-Mazur computability, but not vice versa. For an overview of the logical relations between different notions of computability we again refer to [24] and the introductory textbook [19].

Definition 4. A function $f : \mathbb{R}_c \rightarrow \mathbb{R}_c$ 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.

If we compare the different notions of computability, we immediately see that any computable continuous function is also Banach-Mazur computable, since Definition 4 is the same as the first condition in Definition 3. However, there are infinitely many examples of Banach-Mazur computable functions that are not computable continuous, cf. for example [24] for a detailed discussion. Such functions do not satisfy the second condition in Definition 3 and, accordingly, it is not possible to compute the local variations of these functions.

We further need the concepts of a recursive set and a recursively enumerable set as defined e.g. in [28].

Definition 5. 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 6. A set $\mathcal{A} \subset \mathbb{N}$ is *recursively enumerable* if there exists a recursive function whose domain is exactly \mathcal{A} .

We have the following properties; cf. for example [28]

- \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 $f : \mathbb{N} \rightarrow \mathcal{A}^c$ with $[f(\mathbb{N})] = \mathcal{A}^c$.

2.2. Computable Sequences of Numbers and Functions

In the following we establish some properties of computable sequences which will be needed subsequently.

Theorem 7. *Let $\{x_n^{(1)}\}_{n \in \mathbb{N}}$ and $\{x_n^{(2)}\}_{n \in \mathbb{N}}$ be two computable sequences of computable real numbers with*

$$x_n^{(1)} \leq x_{n+1}^{(1)} \quad \text{and} \quad x_n^{(2)} \geq x_{n+1}^{(2)}, \quad n \in \mathbb{N},$$

and

$$\lim_{n \rightarrow \infty} x_n^{(1)} = \lim_{n \rightarrow \infty} x_n^{(2)} =: x_*.$$

Then x_* is a computable real number, i.e., $x_* \in \mathbb{R}_c$.

Proof. If $\{x_n^{(1)}\}_{n \in \mathbb{N}}$ and $\{x_n^{(2)}\}_{n \in \mathbb{N}}$ are computable sequences of rational numbers, then the result can be found in [25]. The proof can be extended to computable real numbers as follows.

Since $\{x_n^{(1)}\}_{n \in \mathbb{N}}$ is a computable sequence of computable real numbers, there is a computable sequence $\{\varphi_n^{(1)}\}_{n \in \mathbb{N}}$ such that for all $N \in \mathbb{N}$ there exists a computable double sequence $\{a_{n,m}^{(1)}\}_{n,m \in \mathbb{N}}$ with

$$\left| x_n^{(1)} - a_{n,m}^{(1)} \right| < \frac{1}{2^N} \quad \text{for all } m \geq \varphi_n^{(1)}(N).$$

For $m_n = \varphi_n^{(1)}(n)$ we set $a_n^{(1)} = a_{n,m_n}^{(1)}$ so that $\{a_n^{(1)}\}_{n \in \mathbb{N}}$ is a computable sequence of rational numbers and we have

$$x_n^{(1)} > a_n^{(1)} - \frac{1}{2^n}.$$

We set $c_n = \max_{1 \leq i \leq n} [a_i^{(1)} - \frac{1}{2^i}]$ to obtain the sequence $\{c_n\}_{n \in \mathbb{N}}$ which is a computable sequence of rational numbers with

$$c_n \leq c_{n+1}, \quad n \in \mathbb{N},$$

and

$$c_n \leq x_n^{(1)} \leq x_*, \quad n \in \mathbb{N}.$$

Further, we have

$$\begin{aligned} |x_* - c_n| &= |x_* - a_n^{(1)} + a_n^{(1)} - c_n| \\ &\leq |x_* - a_n^{(1)}| + |a_n^{(1)} - c_n| \\ &= |x_* - x_n^{(1)} + x_n^{(1)} - a_n^{(1)}| + |a_n^{(1)} - c_n| \\ &\leq |x_* - x_n^{(1)}| + |x_n^{(1)} - a_n^{(1)}| + |a_n^{(1)} - c_n| \\ &\leq |x_* - x_n^{(1)}| + \frac{1}{2^n} + \frac{1}{2^n} \end{aligned}$$

so that

$$\lim_{n \rightarrow \infty} |x_* - c_n| = 0,$$

i.e., the monotonically increasing computable sequence $\{c_n\}_{n \in \mathbb{N}}$ of rational numbers converges to x_* .

In a similar way, based on the computable sequence $\{x_n^{(2)}\}_{n \in \mathbb{N}}$ we can construct a monotonically decreasing computable sequence $\{d_n\}_{n \in \mathbb{N}}$ of rational numbers with

$$\lim_{n \rightarrow \infty} |x_* - d_n| = 0.$$

Now, we can apply the corresponding result from [25] for computable sequences of rational numbers to conclude that x_* must be a computable real number, i.e., $x_* \in \mathbb{R}_c$. \square

This allows us to prove the following result.

Theorem 8. *Let $\{x_n\}_{n \in \mathbb{N}}$ be a monotonically increasing computable sequence of computable real numbers and let x_* be its limit. If $x_* \in \mathbb{R}_c$, then there exists a recursive function $\varphi : \mathbb{N} \rightarrow \mathbb{N}$ such that for all $N \in \mathbb{N}$ we have for all $n \geq \varphi(N)$*

$$|x_* - x_n| < \frac{1}{2^N}.$$

Proof. For computable sequences of rational numbers, the result can be found in [25]. The proof can be extended to computable real numbers as follows.

We make use of the construction in the proof of Theorem 7 to prove the desired result. Applying this construction to $\{x_n\}_{n \in \mathbb{N}}$ results in a monotonically increasing computable sequence $\{c_n\}_{n \in \mathbb{N}}$ of rational numbers with

$$c_n \leq x_n \leq x_*, \quad n \in \mathbb{N}.$$

Since the result holds for monotonically increasing computable sequences of rational numbers, there exists a recursive function $\varphi : \mathbb{N} \rightarrow \mathbb{N}$ such that for all $N \in \mathbb{N}$ we have for all $n \geq \varphi(N)$

$$0 \leq x_* - x_n \leq x_* - c_n < \frac{1}{2^n}$$

so that

$$|x_* - x_n| < \frac{1}{2^N} \quad \text{for all } n \geq \varphi(N).$$

Thus, the computable sequence of computable real numbers converges effectively to x_* proving the desired result. \square

Remark 9. Note that it is possible to find a computable sequence $\{x_n\}_{n \in \mathbb{N}}$ of rational numbers that converges to a computable real number $x_* \in \mathbb{R}_c$ (which can further be rational), i.e.,

$$\lim_{n \rightarrow \infty} |x_* - x_n| = 0,$$

but the convergence is not effective. According to the following Theorem 10, this sequence is then not monotonically increasing or decreasing.

Next, we establish similar results for computable sequences of computable continuous functions.

Theorem 10. *Let $F : [0, 1] \rightarrow \mathbb{R}$ be a computable continuous function and $\{F_N\}_{N \in \mathbb{N}}$ be a computable sequence thereof with $F_N(x) \leq F_{N+1}(x)$, $x \in [0, 1]$, and*

$$\lim_{N \rightarrow \infty} F_N(x) = F(x).$$

Then there exists a recursive function $\varphi : \mathbb{N} \rightarrow \mathbb{N}$ such that for all $M \in \mathbb{N}$ we have for all $N \geq \varphi(M)$

$$|F(x) - F_N(x)| < \frac{1}{2^M}.$$

Proof. Let $Q_N(x) = F(x) - F_N(x)$, $x \in [0, 1]$. We have $0 \leq Q_{N+1}(x) \leq Q_N(x)$ and $\lim_{N \rightarrow \infty} Q_N(x) = 0$, $x \in [0, 1]$. Let $M \in \mathbb{N}$ be arbitrary. There exists an $N_0 = N_0(M, x)$ with

$$Q_N(x) < \frac{1}{2^M} \quad \text{for all } N \geq N_0(M, x).$$

We define the set

$$\mathcal{S}_{N,M} = \left\{ x \in [0, 1] : Q_N(x) < \frac{1}{2^M} \right\}$$

and observe that $\mathcal{S}_{N,M} \subset \mathcal{S}_{N+1,M}$. Now, $\{\mathcal{S}_{N,M}\}$ is a family of open sets with $[0, 1] \subset \bigcup_{N=1}^{\infty} \mathcal{S}_{N,M}$. Since $[0, 1]$ is a compact set [29], there exists an $N_0(M)$ with $[0, 1] \subset \mathcal{S}_{N_0,M}$ and therewith $Q_{N_0}(x) < \frac{1}{2^M}$ for N_0 and also all $N \geq N_0$. Let

$$\max_{x \in [0,1]} Q_N(x) = C_N.$$

Since Q_N is a computable continuous function, we always have $C_N \in \mathbb{R}_c$. Further, since $\{Q_N\}_{N \in \mathbb{N}}$ is a computable sequence of computable real numbers, the sequence $\{C_N\}_{N \in \mathbb{N}}$ is also a computable sequence of computable real numbers. For all $N \in \mathbb{N}$ it holds that $C_N \geq C_{N+1}$ and

$$\lim_{N \rightarrow \infty} C_N = 0.$$

Accordingly, there exists a recursive function $\varphi : \mathbb{N} \rightarrow \mathbb{N}$ such that for all $M \in \mathbb{N}$ we have for all $N \geq \varphi(M)$

$$|F(x) - F_N(x)| = |Q_N(x)| < \frac{1}{2^M}$$

which proves the desired result. \square

Some remarks are in order:

- 1) The result extends to functions on compact spaces.
- 2) The result remains true for monotonically decreasing functions.
- 3) It is important that F is a computable continuous function. Already for computable sequences of rational numbers with $x_n \leq x_{n+1}$ that converge to a $x_* \notin \mathbb{R}_c$, we do not have effective convergence, see e.g. [30].
- 4) A part of the proof is not effective as we required compactness which is needed to show uniform convergence. This is subsequently used to show the effective convergence of the computable continuous function F .

We can use Theorem 10 to show the following result.

Corollary 11. *Let $\{F_N\}_{N \in \mathbb{N}}$ and $\{G_N\}_{N \in \mathbb{N}}$ be computable sequences of computable continuous functions on $[0, 1]$ with*

$$F_N(x) \leq F_{N+1}(x) \leq G_{N+1}(x) \leq G_N(x)$$

and

$$\lim_{N \rightarrow \infty} F_N(x) = \lim_{N \rightarrow \infty} G_N(x) =: \Phi(x), \quad x \in [0, 1].$$

Then $\Phi : [0, 1] \rightarrow \mathbb{R}$ is also a computable continuous function and $\{F_N\}_{N \in \mathbb{N}}$ and $\{G_N\}_{N \in \mathbb{N}}$ converge effectively to Φ .

Proof. We set

$$Q_N(x) = G_N(x) - F_N(x), \quad x \in [0, 1],$$

and $\{Q_N\}_{N \in \mathbb{N}}$ is a computable sequence of computable continuous functions. For $x \in [0, 1]$ we have

$$Q_N(x) \geq G_{N+1}(x) - F_N(x) \geq G_{N+1}(x) - F_{N+1}(x) = Q_{N+1}(x)$$

and

$$\lim_{N \rightarrow \infty} Q_N(x) = 0, \quad x \in [0, 1].$$

Now, from Theorem 10 follows that the computable sequence $\{Q_N\}_{N \in \mathbb{N}}$ of computable continuous functions converges effectively to zero proving the desired result. \square

We obtain a similar result for computable sequences of Banach-Mazur computable functions.

Theorem 12. *Let $\{F_N\}_{N \in \mathbb{N}}$ and $\{G_N\}_{N \in \mathbb{N}}$ be computable sequences of functions $F_N : [0, 1] \cap \mathbb{R}_c \rightarrow \mathbb{R}_c$ and $G_N : [0, 1] \cap \mathbb{R}_c \rightarrow \mathbb{R}_c$, $N \in \mathbb{N}$, with*

$$\begin{aligned} F_N(x) &\leq F_{N+1}(x), & x \in [0, 1] \cap \mathbb{R}_c, \\ G_N(x) &\geq G_{N+1}(x), & x \in [0, 1] \cap \mathbb{R}_c, \end{aligned}$$

and

$$\lim_{N \rightarrow \infty} F_N(x) = \lim_{N \rightarrow \infty} G_N(x) =: \Phi(x), \quad x \in [0, 1] \cap \mathbb{R}_c.$$

Then $\Phi : [0, 1] \cap \mathbb{R}_c \rightarrow \mathbb{R}$ is also a Banach-Mazur computable function.

Proof. The function $\Phi : [0, 1] \cap \mathbb{R}_c \rightarrow \mathbb{R}$ is well defined. For $x \in [0, 1] \cap \mathbb{R}_c$, the function value $\Phi(x)$ is the limit of the monotonically increasing computable sequence $\{F_N(x)\}_{N \in \mathbb{N}}$ of computable real numbers as well as the limit of the monotonically decreasing computable sequence $\{G_N(x)\}_{N \in \mathbb{N}}$ of computable real numbers. Therefore, we have $\Phi(x) \in \mathbb{R}_c$.

We have to show that for every computable sequence $\{x_n\}_{n \in \mathbb{N}}$ of computable real numbers, the sequence $\{\Phi(x_n)\}_{n \in \mathbb{N}}$ is a computable sequence of computable real numbers as well. Let $y_n = \Phi(x_n)$, $n \in \mathbb{N}$. Similarly as in the proofs of Theorems 7 and 8, there exist computable double sequences $\{\bar{y}_{n,N}\}_{n \in \mathbb{N}, N \in \mathbb{N}}$ and $\{\underline{y}_{n,N}\}_{n \in \mathbb{N}, N \in \mathbb{N}}$ of rational numbers with

$$\bar{y}_{n,N} = G_N(x_n) \quad \text{and} \quad \underline{y}_{n,N} = F_N(x_n),$$

which satisfy the following properties: For every $n \in \mathbb{N}$ and $N \in \mathbb{N}$ it holds

$$\bar{y}_{n,N} \geq \bar{y}_{n,N+1} \quad \text{and} \quad \underline{y}_{n,N} \leq \underline{y}_{n,N+1}$$

and further

$$\lim_{N \rightarrow \infty} \bar{y}_{n,N} = \lim_{N \rightarrow \infty} \underline{y}_{n,N} = y_n.$$

As in the proof of Theorem 7, for $n \in \mathbb{N}$ let for $M \in \mathbb{N}$, $\varphi_n(M)$ be the smallest natural number N such that

$$0 \leq \bar{y}_{n,N} - \underline{y}_{n,N} < \frac{1}{2M}.$$

Then, φ_n is a recursive function and $\{\varphi_n\}_{n \in \mathbb{N}}$ is a computable sequence of recursive functions. From the s-m-n-Theorem [28] follows that there exists also a recursive function $\varphi : \mathbb{N}^2 \rightarrow \mathbb{N}$ with

$$\varphi(n, M) = \varphi_n(M), \quad (n, M) \in \mathbb{N}^2.$$

As in the proof of Theorem 8 this implies that for all $n \in \mathbb{N}$ it holds: For all $M \in \mathbb{N}$ we have for all $N \geq \varphi(n, M)$

$$|y_n - \underline{y}_{n,N}| < \frac{1}{2M},$$

i.e., $\{y_n\}_{n \in \mathbb{N}}$ is a computable sequence of computable real numbers which completes the proof. \square

It is clear that this result also applies to computable sequences of Borel computable functions. Also in this case, the function Φ must be Banach-Mazur computable.

In the following, we will use these results and in particular Theorem 10, Theorem 12, and Corollary 11 to study the computability of the capacity of FSCs.

3. Finite State Channels

In this section we introduce the concept of finite state channels which are suitable to model discrete channels with memory [1, 8, 12].

3.1. Basic Definitions

Let \mathcal{X} , \mathcal{Y} , and \mathcal{S} be finite input, output, and state sets. FSCs are usually specified by its underlying probability law

$$(3) \quad p(y_n, s_n | x_n, s_{n-1}) \in \mathcal{P}(\mathcal{Y} \times \mathcal{S} | \mathcal{X} \times \mathcal{S})$$

where $y_n \in \mathcal{Y}$ and $s_n \in \mathcal{S}$ are the output and state of the channel at time instant n whose probability depend on the input $x_n \in \mathcal{X}$ at time instant n and on the previous state $s_{n-1} \in \mathcal{S}$ at time instant $n - 1$.

We assume that the output Y_n and state S_n are statistically independent given x_n and s_{n-1} so that (3) can be written as

$$(4) \quad p(y_n, s_n | x_n, s_{n-1}) = p(y_n | x_n, s_{n-1})q(s_n | x_n, s_{n-1})$$

for $p(y_n | x_n, s_{n-1}) \in \mathcal{P} := \mathcal{P}(\mathcal{Y} | \mathcal{X} \times \mathcal{S})$ and $q(s_n | x_n, s_{n-1}) \in \mathcal{Q} := \mathcal{P}(\mathcal{S} | \mathcal{X} \times \mathcal{S})$. The corresponding sets of computable conditional probabilities are denoted by $\mathcal{P}_c := \mathcal{P}_c(\mathcal{Y} | \mathcal{X} \times \mathcal{S})$ and $\mathcal{Q}_c := \mathcal{P}_c(\mathcal{S} | \mathcal{X} \times \mathcal{S})$, respectively.

Remark 13. Not that the assumption of independence of Y_n and S_n and its consequence on the probability law as shown in (4) will be no loss of generality. In the end, we will show that already the special class (4) of FSCs is not Turing computable so that this must be the case for the general class (3) as well.

In general, $p^n(y^n | x^n)$ for block length n is undefined for a FSC and we have to consider the general $p^n(y^n, s_n | x^n, s_0)$ which is the probability of the output sequence y^n and a final state s_n at time instant n given an input

sequence x^n and an initial state s_0 . This can be calculated inductively from

$$(5) \quad p^n(y^n, s_n | x^n, s_0) = \sum_{s_{n-1} \in \mathcal{S}} p(y_n, s_n | x_n, s_{n-1}) p^{n-1}(y^{n-1}, s_{n-1} | x^{n-1}, s_0),$$

cf. [1]. Further, by summing over the final state we obtain

$$(6) \quad p^n(y^n | x^n, s_0) = \sum_{s_n \in \mathcal{S}} p^n(y^n, s_n | x^n, s_0).$$

Definition 14. An (n, M) -code for an FSC consists of an encoder $f : \mathcal{M} \times \mathcal{S} \rightarrow \mathcal{X}^n$ that maps the message $m \in \mathcal{M} = \{1, \dots, M\}$ and the initial state $s_0 \in \mathcal{S}$ into the codeword $x^n \in \mathcal{X}^n$, and a decoder $\varphi : \mathcal{Y}^n \times \mathcal{S} \rightarrow \mathcal{M}$ that estimates the transmitted message $\hat{m} \in \mathcal{M}$ based on the received output $y^n \in \mathcal{Y}^n$ and the initial state $s_0 \in \mathcal{S}$.

For the initial state $s_0 \in \mathcal{S}$ the average probability of error of such a code based on (6) is

$$\bar{e}(s_0) = \frac{1}{|\mathcal{M}|} \sum_{m \in \mathcal{M}} \sum_{y^n : \varphi(y^n, s_0) \neq m} p(y^n | f(m, s_0), s_0).$$

Definition 15. A rate $R > 0$ is an *achievable rate* for an FSC if for all $\tau > 0$ there exists an $n(\tau) \in \mathbb{N}$ and a sequence of (n, M) -codes such that for all $n \geq n(\tau)$ we have $\frac{1}{n} \log M > R - \tau$ and $\bar{e}(s_0) \leq \lambda_n$ for $s_0 \in \mathcal{S}$ with $\lambda_n \rightarrow 0$ as $n \rightarrow \infty$. The *capacity* C of an FSC is given by the supremum of all achievable rates R .

The capacity C of an FSC is a function of the communication parameters $p \in \mathcal{P}$ and $q \in \mathcal{Q}$, cf. (4), as well as the initial state $s_0 \in \mathcal{S}$. Accordingly, we write $C = C(\{p, q, s_0\})$.

3.2. General Capacity Formulas

We will study the computability of the capacity function C in dependence on the communication parameters $\{p, q, s_0\}$. These will be the inputs to the corresponding Turing machine. For this purpose, we need a corresponding expression for $C(\{p, q, s_0\})$ as for example the general formula provided by Verdú and Han in [16]. For the FSC as defined above, the capacity can be

expressed in a multi-letter form as

$$(7) \quad C(\{p, q, s_0\}) = \lim_{n \rightarrow \infty} \sup_{X^n} \frac{1}{n} I(X^n; Y^n | s_0)$$

according to the underlying probability law (5)-(6). This has been shown to be valid for information stable channels [31], but does not hold in full generality. Moreover, this expression cannot be computed immediately as it is the limit of a sequence of optimization problems. Furthermore, it is not even clear if $C(\{p, q, s_0\})$ is a computable real number for computable p and q . Another formula for the capacity based on the inf-information rate has been established in [16]

$$(8) \quad C(\{p, q, s_0\}) = \sup_{\mathbf{X}} \underline{I}(\mathbf{X}; \mathbf{Y})$$

where $\underline{I}(\mathbf{X}; \mathbf{Y})$ is the inf-information rate as defined in [32]. In general, this expression cannot be evaluated easily.

For the special class of so-called indecomposable channels, there exists a simple capacity expression for $C(\{p, q, s_0\})$. This is discussed next.

3.3. Indecomposable Channels

The class of IFSCs goes back to [12] and refers to those FSCs for which the effect of the initial state vanishes with time. For the precise definition of this, we follow [1, Sec. 4] and set $q^n(s_n | x^n, s_0) = \sum_{y^n \in \mathcal{Y}^n} p^n(y^n, s_n | x^n, s_0)$.

Definition 16. An FSC is called *indecomposable* if for every $\epsilon > 0$ there exists an $n_0 \in \mathbb{N}$ such that for all $n \geq n_0$ we have $|q^n(s_n | x^n, s_0) - q^n(s_n | x^n, s'_0)| \leq \epsilon$ for all $s_n \in \mathcal{S}$, $x^n \in \mathcal{X}^n$, $s_0 \in \mathcal{S}$, and $s'_0 \in \mathcal{S}$.

For the capacity of IFSCs, we need the functions:

$$(9a) \quad \underline{C}_n(\{p, q\}) = \frac{1}{n} \max_{X^n} \min_{s_0} I(X^n; Y^n | s_0)$$

$$(9b) \quad \overline{C}_n(\{p, q\}) = \frac{1}{n} \max_{X^n} \max_{s_0} I(X^n; Y^n | s_0).$$

Remark 17. Note that for fixed $n \in \mathbb{N}$ and computable parameters $\{p, q, s_0\} \in \mathcal{P}_c \times \mathcal{Q}_c \times \mathcal{S}$, the functions \underline{C}_n and \overline{C}_n in (9) are computable functions, i.e., we have $\underline{C}_n : \mathcal{P}_c \times \mathcal{Q}_c \times \mathcal{S} \rightarrow \mathbb{R}_c$ and $\overline{C}_n : \mathcal{P}_c \times \mathcal{Q}_c \times \mathcal{S} \rightarrow \mathbb{R}_c$.

The sequences $\{\underline{C}_n\}_{n=1}^\infty$ and $\{\overline{C}_n\}_{n=1}^\infty$ for $\{p, q, s_0\} \in \mathcal{P} \times \mathcal{Q} \times \mathcal{S}$ converge and we have

$$(10a) \quad \underline{C}(\{p, q\}) = \lim_{n \rightarrow \infty} \underline{C}_n(\{p, q\})$$

$$(10b) \quad \overline{C}(\{p, q\}) = \lim_{n \rightarrow \infty} \overline{C}_n(\{p, q\})$$

which are also called *lower capacity* and *upper capacity*, respectively. If the FSC is indecomposable, lower and upper capacities coincide and are equal to the capacity, i.e., $\underline{C}(\{p, q\}) = \overline{C}(\{p, q\}) = C(\{p, q, s_0\})$.

3.4. Main Problem Formulation

For fixed alphabets \mathcal{X} , \mathcal{Y} , and \mathcal{S} , the capacity C of an FSC is a function of the underlying system parameters $\{p, q, s_0\}$. The previous discussion leads to the following questions of interest:

Question 1: Is the capacity $C(\{p, q, s_0\})$ for fixed initial state $s_0 \in \mathcal{S}$ a computable continuous function on \mathcal{P} and \mathcal{Q} ?

Here, we allow arbitrary $p \in \mathcal{P}$ and $q \in \mathcal{Q}$ as inputs for the capacity function C . From Corollary 11 we already see that this question is naturally connected to the question whether or not it is possible to find computable continuous lower and upper bounds on the capacity. Lower bounds originate from actual coding schemes and general achievability results, while the upper bounds are established via converse arguments.

From a practical point of view, such lower and upper bounds should be computable to enable a numerical evaluation on digital computers. Therefore, it is reasonable to study the capacity C as a function on computable inputs $p \in \mathcal{P}_c$ and $q \in \mathcal{Q}_c$. This leads to following question:

Question 2: Is the capacity $C(\{p, q, s_0\})$ for fixed initial state $s_0 \in \mathcal{S}$ a Borel computable function (and therewith Turing computable)?

While Borel computability is a strong notion of computability, Banach-Mazur computability is considered to be the weakest form of computability and it is of interest to pose a similar question for this notion as follows:

Question 3: Is the capacity $C(\{p, q, s_0\})$ for fixed initial state $s_0 \in \mathcal{S}$ a Banach-Mazur computable function?

As for the first question, the lower and upper bounds on the capacity in Questions 2 and 3 should be algorithmically computable.

Question 4: Is the capacity $C(\{p, q, s_0\})$ for fixed initial state $s_0 \in \mathcal{S}$ approximately Turing computable?

Remark 18. In the following, we will provide negative answers to Questions 1-3. As the capacity is shown to be non-computable, Question 4 about whether or not the capacity is at least approximately computable becomes particularly relevant. To make sure that this question is not trivial, the tolerated approximation error should not be too large. Also for Question 4 we will provide a negative answer for certain approximation errors.

4. Computability Analysis of the FSC Capacity

In this section, we show that the capacity function C is not Banach-Mazur computable and therewith also not Borel and Turing computable. Subsequently, we discuss the case when the capacity of an FSC becomes a computable real number.

4.1. Non-Banach-Mazur Computability

In general, the capacity of an FSC is given by (8) and for every $n \in \mathbb{N}$, the inf-information rate expression $\sup_{\mathbf{X}} \underline{I}(\mathbf{X}; \mathbf{Y})$ is indeed Turing computable (we omit the details due to space constraints). However, in the end the capacity in (8) is given by the limit of for $n \rightarrow \infty$ and the convergence of this limit need not be effective and uniform on $\{p, q, s_0\} \in \mathcal{P}_c \times \mathcal{Q}_c \times \mathcal{S}$, i.e., for a given $\epsilon \in \mathbb{Q}$, e.g., $\epsilon^* = \frac{1}{\mu}$ with $\mu \in \mathbb{N}$, we cannot algorithmically compute when $|f_n(p, q, s_0) - C(\{p, q, s_0\})| < \epsilon$ is satisfied.

Theorem 19. *For all $|\mathcal{X}| \geq 2$, $|\mathcal{Y}| \geq 2$, and $|\mathcal{S}| \geq 2$, the capacity function $C(\{p, q, s_0\}) : \mathcal{P}_c \times \mathcal{Q}_c \times \mathcal{S} \rightarrow \mathbb{R}$ of the FSC with parameters $\{p, q, s_0\}$ is not Banach-Mazur computable.*

Proof. We first prove the result for $|\mathcal{X}| = |\mathcal{Y}| = |\mathcal{S}| = 2$ and subsequently outline how it extends to the general case.

If the finite state channel $\{p, q, s_0\}$, $s_0 \in \mathcal{S} = \{0, 1\}$, is indecomposable, then the effect of the initial state vanishes and we have

$$C(\{p, q, 0\}) = C(\{p, q, 1\}) = \overline{C}(\{p, q\}) = \underline{C}(\{p, q\})$$

and further

$$\underline{C}(\{p, q\}) = \min_{s_0 \in \{0, 1\}} C(\{p, q, s_0\}) \leq \max_{s_0 \in \{0, 1\}} C(\{p, q, s_0\}) = \overline{C}(\{p, q\}).$$

Next we consider the channel

$$(11) \quad p(y_n|x_n, 0) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad p(y_n|x_n, 1) = \begin{pmatrix} 1-\epsilon & \epsilon \\ \epsilon & 1-\epsilon \end{pmatrix}$$

for some $0 < \epsilon < 1/2$, i.e., for state $s_{n-1} = 0$ the channel is noiseless, while for $s_{n-1} = 1$ it is noisy. Further, we consider the state distribution

$$(12) \quad \hat{q}(s_n|x_n, 0) = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad \hat{q}(s_n|x_n, 1) = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

to be independent of $x_n \in \mathcal{X}$ so that for $s_n \in \mathcal{S}$ and $s_{n-1} \in \mathcal{S}$ arbitrary we have

$$(13) \quad \hat{q}(s_n|x_n, s_{n-1}) = \hat{q}(s_n|s_{n-1}).$$

Note that p and \hat{q} as defined above are computable, i.e., we have $p \in \mathcal{P}_c := \mathcal{P}_c(\mathcal{Y}|\mathcal{X} \times \mathcal{S})$ and $\hat{q} \in \mathcal{Q}_c := \mathcal{P}_c(\mathcal{S}|\mathcal{X} \times \mathcal{S})$. In what follows, we consider the finite state channel $\{p, \hat{q}, s_0\}$, $s_0 \in \{0, 1\}$, as defined above.

We observe that $\{p, \hat{q}, 0\}$ is given by a simple discrete memoryless channel (DMC) $p(y|x, 0)$, $x \in \mathcal{X}$, $y \in \mathcal{Y}$, since the state is always $s_n = 0$ due to the definition of \hat{q} , cf. (12). Accordingly, the capacity is $C(\{p, \hat{q}, 0\}) = 1$ in this case, since the alphabets are binary and the channel is noiseless.

We further observe that $\{p, \hat{q}, 1\}$ corresponds to the DMC $p(y|x, 1)$, $x \in \mathcal{X}$, $y \in \mathcal{Y}$, i.e., it is a binary symmetric channel (BSC). The optimal input distribution is known to be the uniform distribution and the capacity in this case is then $C(\{p, \hat{q}, 1\}) = C_{\text{BSC}}(\epsilon) = 1 - H_2(\epsilon) < 1$.

Next, we show that both functions $C(\{p, q, 0\})$ and $C(\{p, q, 1\})$ cannot be simultaneously Banach-Mazur computable. For this purpose, we take an arbitrary recursively enumerable, but not recursive, set $\mathcal{A} \subset \mathbb{N}$. Let $\mathfrak{T}_{\mathcal{A}}$ be a Turing machine that stops if and only if for input n we have $n \in \mathcal{A}$. Otherwise, $\mathfrak{T}_{\mathcal{A}}$ runs forever. Such a Turing machine can easily be found as

argued next: Let $\varphi_{\mathcal{A}} : \mathbb{N} \rightarrow \mathbb{N}$ be a recursive function that lists all elements of the set \mathcal{A} and for which $\varphi_{\mathcal{A}} : \mathbb{N} \rightarrow \mathcal{A}$ is a unique function.

Let $n \in \mathbb{N}$ be arbitrary. The Turing machine $\mathfrak{T}_{\mathcal{A}}$ with input n is defined as follows: We start with $l = 1$ and compute $\varphi_{\mathcal{A}}(1)$. If $n = \varphi_{\mathcal{A}}(1)$, then the Turing machine stops. In the other case, the Turing machine computes $\varphi_{\mathcal{A}}(2)$. Similarly, if $n = \varphi_{\mathcal{A}}(2)$, then the Turing machine stops and otherwise, it continues computing the next element. It is clear that this Turing machine stops if and only if $n \in \mathcal{A}$.

Assume that both functions $C(\{p, q, 0\})$ and $C(\{p, q, 1\})$ are Banach-Mazur computable. For $\lambda \in [0, \frac{1}{2}] \cap \mathbb{R}_c$ we consider

$$q_{\lambda}(s_n|x_n, 0) = \begin{pmatrix} 1 - \lambda \\ \lambda \end{pmatrix} \quad \text{and} \quad q_{\lambda}(s_n|x_n, 1) = \begin{pmatrix} \lambda \\ 1 - \lambda \end{pmatrix}.$$

Of course, for $\lambda \in [0, \frac{1}{2}] \cap \mathbb{R}_c$, $q_{\lambda}(s_n|x_n, 0)$ and $q_{\lambda}(s_n|x_n, 1)$ are computable probability distributions. Let

$$\begin{aligned} q_0(s_n|x_n, 0) &= \hat{q}(s_n|x_n, 0), & s_n \in \mathcal{S}, x_n \in \mathcal{X}, \\ q_0(s_n|x_n, 1) &= \hat{q}(s_n|x_n, 1), & s_n \in \mathcal{S}, x_n \in \mathcal{X}. \end{aligned}$$

We have

$$C(\{p, q_0, 1\}) - C(\{p, q_0, 0\}) = 1 - (1 - H_2(\epsilon)) = H_2(\epsilon) > 0.$$

For $0 < \lambda \leq \frac{1}{2}$ the FSC $\{p, q_{\lambda}, s_0\}$ $s_0 \in \mathcal{S}$ is indecomposable and therewith we have

$$C(\{p, q_{\lambda}, 0\}) = C(\{p, q_{\lambda}, 1\}).$$

Now, for every $n \in \mathbb{N}$ and $m \in \mathbb{N}$ let

$$\lambda_{n,m} = \begin{cases} \frac{1}{2^l} & \mathfrak{T}_{\mathcal{A}} \text{ stops for input } n \text{ after } l \leq m \text{ steps} \\ \frac{1}{2^m} & \mathfrak{T}_{\mathcal{A}} \text{ does not stop for input } n \text{ after } m \text{ steps.} \end{cases}$$

Then the sequence $\{\lambda_{n,m}\}_{n,m \in \mathbb{N}}$ is a computable double sequence of rational numbers. For arbitrary $n \in \mathbb{N}$ and arbitrary $m, m_1 \in \mathbb{N}$, $m \geq m_1$, it holds

$$(14) \quad |\lambda_{n,m} - \lambda_{n,m_1}| = |\lambda_{n,m_1} - \lambda_{n,m}| = \lambda_{n,m_1} - \lambda_{n,m} < \frac{1}{2^{m_1}}$$

since if the Turing machine $\mathfrak{T}_{\mathcal{A}}$ has stopped for input n for $l \leq m_1$, then $\lambda_{n,m_1} = \lambda_{n,m}$ and (14) is trivially satisfied. If the Turing machine $\mathfrak{T}_{\mathcal{A}}$ has not stopped for input n after m_1 iterations, then $\lambda_{n,m_1} = \lambda_{n,m} = \frac{1}{2^{m_1}} - \lambda_{n,m} <$

$\frac{1}{2^{m-1}}$, since $\lambda_{n,m} > 0$ for all $n \in \mathbb{N}$, so that (14) is satisfied as well. Accordingly, we observe that $\{\lambda_{n,m}\}_{m \in \mathbb{N}}$ is a sequence that converges effectively and there exists one $\lambda_n^* \in \mathbb{R}_c$ with

$$\lim_{m \rightarrow \infty} |\lambda_n^* - \lambda_{n,m}| = 0.$$

Furthermore, since $\{\lambda_{n,m}\}_{n,m \in \mathbb{N}}$ is computable double sequence, the sequence $\{\lambda_n^*\}_{n \in \mathbb{N}}$ is a computable sequence of computable real numbers. It further holds $\lambda_n^* \geq 0$ with equality if and only if the Turing machine $\mathfrak{T}_{\mathcal{A}}$ does not stop for input n .

Since $C(\{p, q, 0\})$ and $C(\{p, q, 1\})$ are assumed to be Banach-Mazur computable functions, the difference $\Phi(\{p, q\}) = C(\{p, q, 1\}) - C(\{p, q, 0\})$ is a Banach-Mazur computable function as well. Then, the sequence $\{\mu_n\}_{n \in \mathbb{N}}$ with

$$\mu_n = \Phi(\{p, q, \lambda_n^*\}), \quad n \in \mathbb{N},$$

is a computable sequence of computable real numbers. With this, we find a computable double sequence $\{\nu_{n,m}\}_{n,m \in \mathbb{N}}$ of rational numbers with

$$|\mu_n - \nu_{n,m}| < \frac{1}{2^m}.$$

For every n , we can consider the following Turing machine \mathfrak{T}_* : For input n , we set $m = 1$ and check if

$$\nu_{n,1} > \frac{1}{2}$$

is satisfied. If this is true, the Turing machine stops. Otherwise, we set $m = 2$ and check if

$$\nu_{n,2} > \frac{1}{4}$$

is satisfied. If this is true, the Turing machine stops. Otherwise, it continues as described. Next, we show that this Turing machine \mathfrak{T}_* stops for input n if and only if $\mu_n > 0$.

“ \Leftarrow ” If $\mu_n > 0$, then there exists an m_0 with

$$\frac{1}{2^{m_0}} < \frac{\mu_n}{2}$$

so that

$$\begin{aligned} \mu_n &= \mu_n - \nu_{n,m_0} + \nu_{n,m_0} \leq |\mu_n - \nu_{n,m_0}| + \nu_{n,m_0} \\ &< \frac{1}{2^{m_0}} + \nu_{n,m_0} < \frac{\mu_n}{2} + \nu_{n,m_0}, \end{aligned}$$

i.e., the Turing machine \mathfrak{T}_* stops for input n within m_0 steps.

“ \Rightarrow ” It holds $\nu_{n,\hat{m}} > \frac{1}{2^{\hat{m}}}$ for a certain \hat{m} . Then,

$$\begin{aligned} \frac{1}{2^{\hat{m}}} &< \nu_{n,\hat{m}} = \nu_{n,\hat{m}} - \mu_n + \mu_n \\ &\leq |\nu_{n,\hat{m}} - \mu_n| + \mu_n < \frac{1}{2^{\hat{m}}} + \mu_n \end{aligned}$$

so that $\mu_n > 0$ is true.

Next, for input $n \in \mathbb{N}$, we define the Turing machine $\mathfrak{T}_{\mathcal{S}}$ as follows: We run both previous Turing machines $\mathfrak{T}_{\mathcal{A}}$ and \mathfrak{T}_* in parallel for input n , where each Turing machine operates step by step as discussed above. We have already shown that $\mathfrak{T}_{\mathcal{A}}$ stops for input n if and only if $n \in \mathcal{A}$. Further, we have shown that \mathfrak{T}_* stops for input n if and only if $\mu_n > 0$. This is true if and only if the Turing machine $\mathfrak{T}_{\mathcal{A}}$ does not stop for input n , i.e., whenever $n \in \mathcal{A}^c$. As a consequence, one of these Turing machines must always stop for an input n . We set

$$\mathfrak{T}_{\mathcal{S}}(n) = \begin{cases} \{n \in \mathcal{A}\} & \text{if } \mathfrak{T}_{\mathcal{A}} \text{ stops for input } n \\ \{n \in \mathcal{A}^c\} & \text{if } \mathfrak{T}_* \text{ stops for input } n. \end{cases}$$

With this, we have shown that \mathcal{A} is a recursive set. But this is a contradiction so that the assumption that both functions $C(\{p, q, 0\})$ and $C(\{p, q, 1\})$ are Banach-Mazur computable is wrong. This completes the proof.

For $|\mathcal{X}| \geq 2$, $|\mathcal{Y}| \geq 2$, and $|\mathcal{S}| \geq 2$ arbitrary, we take the sequences of parameters $\{p, \hat{q}\}$ and $\{p, q_k\}$ as above and extend them as follows: We set $p(y|x_n, s_{n-1}) = 0$ for $y \in \mathcal{Y} \setminus \{0, 1\}$, $x_n \in \mathcal{X}$, $s_{n-1} \in \mathcal{S}$ and also for $x_n \in \mathcal{X} \setminus \{0, 1\}$, $s_{n-1} \in \mathcal{S} \setminus \{0, 1\}$ to preserve the above constructed behavior. We do the same for \hat{q} and q_k . We observe that we still have $p \in \mathcal{P}_c$ and $\hat{q}, q_k \in \mathcal{Q}_c$. With this and the previous arguments we can conclude on the same result. \square

Remark 20. This result and implications thereof can further be strengthened for countably infinite state sets. In particular, for computable compound channels with countably infinite state sets, the capacity need not be a computable real number in general, cf. also [33].

Remark 21. The techniques used to prove Theorem 19 can be extended to various channel models and operational (communication) tasks in information theory. For example, the problem of secret key generation with rate-limited public discussion has been studied in [34] and the problem of identification with feedback in [35].

Remark 22. The proof of Theorem 19 provides additional deeper insights. This has been developed in detail in [35] for the identification with feedback capacity. By modifying the proof above, one is able to show the following: It is possible to connect the algorithmic computation of the capacity to hard problems in pure mathematics such as Goldbach's Conjecture and the Riemann Hypothesis. To this end, it is possible to find an oracle Turing machine with the following properties: Given finitely many values of the capacity function of the given computable channel, the oracle Turing machine that gets the capacity value of certain computable FSCs as oracle can immediately prove or disprove Goldbach's Conjecture and the Riemann Hypothesis.

Remark 23. It is not clear if similar results hold for the capacity of time-continuous channels as in [2]. Accordingly, it is not clear if the technique presented above is applicable in this case at all. A more detailed discussion on this is given in Section 6.

In the construction of the proof of Theorem 19 above, we assume the special case in which the current state s_n does not depend on the current input x_n but only on the previous state s_{n-1} . This is the special class of *finite fading channels (FFCs)* that naturally applies to wireless communications where the fading state of the channel is independent of the transmitted signal. We immediately obtain the following corollary.

Corollary 24. *For all $|\mathcal{X}| \geq 2$, $|\mathcal{Y}| \geq 2$, and $|\mathcal{S}| \geq 2$, the capacity function $C(\{p, q, s_0\}) : \mathcal{P}_c \times \mathcal{Q}_c \times \mathcal{S} \rightarrow \mathbb{R}$ of the FFC with parameters $\{p, q, s_0\}$ is not Banach-Mazur computable.*

We see that, in general, the capacity of an FSC is not Banach-Mazur and therewith also not Turing computable. However, for special cases of FSCs the capacity becomes Turing computable as e.g. the zero-error capacity [9] or the feedback capacity [11] of the trapdoor channel; but in general, there is no algorithm that can compute the capacity as a function of the parameters $\{p, q, s_0\}$.

Remark 25. Banach-Mazur computability requires the function to operate on computable reals, cf. Definition 4. In Theorem 19 we have shown that $C(\{\cdot, \cdot, s_0\})$ is not Banach-Mazur computable, but this does not imply that the function $C(\{\cdot, \cdot, s_0\})$ itself is not a mapping from computable probability

distributions to computable reals, i.e.,

$$(15) \quad C(\{\cdot, \cdot, s_0\}) : \mathcal{P}_c \times \mathcal{Q}_c \rightarrow \mathbb{R}_c \quad \text{for all } s_0 \in \mathcal{S}.$$

The problem in showing this, is the following: Although the capacity expression (8) is a multi-letter formula which converges, the speed of convergence does not need to be effective, i.e., it may not be representable by an effectively computable function. And indeed, it is not clear whether or not the convergence of (8) is effective.

Next, we study the existence of computable tight lower and upper bounds on the capacity function. First, we study such bounds which are computable continuous functions on the parameters $\{p, q\}$. As lower and upper bounds should be numerically evaluable, this is a very reasonable requirement, cf. also Remark 2.

Theorem 26. *For $|\mathcal{X}| \geq 2$, $|\mathcal{Y}| \geq 2$, and $|\mathcal{S}| \geq 2$ arbitrary but fixed, there exists an $s_0 \in \mathcal{S}$ such that the following holds: There exists no computable sequences $\{F_N\}_{N \in \mathbb{N}}$ and $\{G_N\}_{N \in \mathbb{N}}$ of computable continuous functions with*

- 1) $F_N : \mathcal{P} \times \mathcal{Q} \rightarrow \mathbb{R}$ and $G_N : \mathcal{P} \times \mathcal{Q} \rightarrow \mathbb{R}$, $N \in \mathbb{N}$,
- 2) $F_N(p, q) \leq C(\{p, q, s_0\})$, $p \in \mathcal{P}$, $q \in \mathcal{Q}$, $N \in \mathbb{N}$, and $\lim_{N \rightarrow \infty} F_N(p, q) = C(\{p, q, s_0\})$ for all $p \in \mathcal{P}$, $q \in \mathcal{Q}$,
- 3) $C(\{p, q, s_0\}) \leq G_N(p, q)$, $p \in \mathcal{P}$, $q \in \mathcal{Q}$, $N \in \mathbb{N}$, and $\lim_{N \rightarrow \infty} G_N(p, q) = C(\{p, q, s_0\})$ for all $p \in \mathcal{P}$, $q \in \mathcal{Q}$.

Proof. The result follows immediately from Corollary 11. If such sequences $\{F_N\}_{N \in \mathbb{N}}$ and $\{G_N\}_{N \in \mathbb{N}}$ would exist, then C would be a computable continuous function which is a contradiction, since C is for a certain $s_0 \in \mathcal{S}$ not Banach-Mazur computable. \square

This result shows that an approximation of C by computable continuous functions is not possible. From this, we can immediately conclude the following.

Corollary 27. *For all computable sequences $\{F_N\}_{N \in \mathbb{N}}$ and $\{G_N\}_{N \in \mathbb{N}}$ of computable continuous functions for which there exists an $s_0 \in \mathcal{S}$ such that*

for $N \in \mathbb{N}$ it holds that

$$F_N(p, q) \leq C(\{p, q, s_0\})$$

for all $p \in \mathcal{P}$ and $q \in \mathcal{Q}$, and for $N \in \mathbb{N}$ it holds that

$$C(\{p, q, s_0\}) \leq G_N(p, q)$$

for all $p \in \mathcal{P}$ and $q \in \mathcal{Q}$, there must exist a $(p_*, q_*) \in \mathcal{P} \times \mathcal{Q}$ such that

$$(16) \quad 0 < \max \left\{ \limsup_{N \rightarrow \infty} |C(\{p_*, q_*, s_0\}) - F_N(p_*, q_*)|, \right. \\ \left. \limsup_{N \rightarrow \infty} |C(\{p_*, q_*, s_0\}) - G_N(p_*, q_*)| \right\}.$$

Proof. These statements follow immediately from Theorem 26, since if (16) would be zero for all $(p, q) \in \mathcal{P} \times \mathcal{Q}$, then this would imply that C is a computable function. \square

As a consequence from this result we can conclude that for the capacity of general FSCs, there is either no computable achievability or no computable converse (or both are non-computable).

The functions $\{F_N\}$ can be interpreted as lower bounds for achievable rates and the capacity respectively. Of course, such bounds should be effectively computable so that they can be numerically evaluated. These bounds should improve with increasing $N \in \mathbb{N}$, i.e., $F_N(p, q) \leq F_{N+1}(p, q)$, $(p, q) \in \mathcal{P} \times \mathcal{Q}$, and further should be asymptotically tight, i.e., for $N \rightarrow \infty$ the sequence $\{F_N\}_{N \in \mathbb{N}}$ should converge pointwise to $C(\{p, q, s_0\})$.

Accordingly, the functions $\{G_N\}$ can be seen as upper bounds on the achievable rates and the capacity respectively. Similarly, it is required that these bounds are effectively computable and further $C(\{p, q, s_0\}) \leq G_{N+1}(p, q) \leq G_N(p, q)$, $(p, q) \in \mathcal{P} \times \mathcal{Q}$, i.e., the bounds should improve with increasing $N \in \mathbb{N}$.

However, Corollary 27 shows that we cannot find such functions $\{F_N\}$ and $\{G_N\}$. Accordingly, it is impossible that both achievability and converse are effectively computable at the same time. As a consequence, one of these must be non-computable so that we cannot find an entropic characterization for the capacity. This also means that there exist computable FSCs for which computable lower and upper bounds can never be simultaneously be tight.

Remark 28. Finally, we note that the results of Theorem 26 and Corollary 27 remain true if the requirement of $\{F_N\}_{N \in \mathbb{N}}$ and $\{G_N\}_{N \in \mathbb{N}}$ being

computable sequences of computable continuous functions is weakened to computable sequences of Banach-Mazur computable sequences.

Note that Corollary 27 further provides a negative answer to Question 4. In particular, the proof of Theorem 26 yields lower bounds for the error, for which the capacity cannot be approximated. Note that the statement of non-approximability is strictly stronger than the statement of non-Turing-computability. Indeed, with the results in [36] it is possible to show that there are channels whose capacity is not Turing computable but are approximable for any given approximation error.

4.2. Capacity being a Computable Real Number

Next, we further study the behavior of the capacity function (15) in more detail and address the question if the capacity value itself is a computable real number, cf. also Remark 25. The following Theorem 29 provides a result for a large class of computable FSCs.

Theorem 29. *For every computable FSC $\{p, q, s_0\}$, $s_0 \in \mathcal{S}$, that satisfies $\overline{C}(\{p, q\}) = \underline{C}(\{p, q\})$, we have $C(\{p, q, s_0\}) \in \mathbb{R}_c$ for all $s_0 \in \mathcal{S}$, i.e., the capacity is a computable real number.*

Proof. We make use of the following properties. Let $p \in \mathcal{P}$ and $q \in \mathcal{Q}$ be arbitrary. Then

$$\begin{aligned}\overline{C}(\{p, q\}) &= \inf_{n \in \mathbb{N}} \left(\overline{C}_n(\{p, q\}) + \frac{\log |\mathcal{S}|}{n} \right), \\ \underline{C}(\{p, q\}) &= \sup_{n \in \mathbb{N}} \left(\underline{C}_n(\{p, q\}) - \frac{\log |\mathcal{S}|}{n} \right),\end{aligned}$$

see [1, Theorem 4.6.1].

For every $n \in \mathbb{N}$ and $\{p, q\} \in \mathcal{P}_c \times \mathcal{Q}_c$, $\overline{C}_n(\{p, q\})$ is a computable number. Accordingly, $\{\overline{C}_n(\{p, q\})\}_{n=1}^{\infty}$ is a computable sequence of computable reals. We define

$$(17) \quad \overline{C}(M; \{p, q\}) := \min_{1 \leq n \leq 2^M} \left(\overline{C}_n(\{p, q\}) + \frac{\log |\mathcal{S}|}{n} \right).$$

We see that $\overline{C}(M; \{p, q\})$ is a computable real for $M \in \mathbb{N}$ and the corresponding sequence $\{\overline{C}(M; \{p, q\})\}_{M=1}^{\infty}$ is a computable sequence of computable reals. We have $\overline{C}(M; \{p, q\}) \geq \overline{C}(M+1; \{p, q\})$ for $M \in \mathbb{N}$, i.e., the sequence

is monotonically decreasing and it holds $\lim_{M \rightarrow \infty} \overline{C}(M; \{p, q\}) = \overline{C}(\{p, q\})$. We further set

$$(18) \quad \underline{C}(M; \{p, q\}) := \max_{1 \leq n \leq 2^M} \left(\underline{C}_n(\{p, q\}) - \frac{\log |\mathcal{S}|}{n} \right)$$

and similarly obtain $\underline{C}(M+1; \{p, q\}) \geq \underline{C}(M; \{p, q\})$ for $M \in \mathbb{N}$. It holds $\lim_{M \rightarrow \infty} \underline{C}(M; \{p, q\}) = \underline{C}(\{p, q\})$. By assumption we further have for all $s_0 \in \mathcal{S}$, $\underline{C}(\{p, q\}) = C(\{p, q, s_0\}) = \overline{C}(\{p, q\})$.

Next, we consider the function

$$g_M(\{p, q\}) := \overline{C}(M; \{p, q\}) - \underline{C}(M; \{p, q\}).$$

Due to the monotonicity of both sequences, we have $0 \leq g_{M+1}(\{p, q\}) \leq g_M(\{p, q\})$ and $\lim_{M \rightarrow \infty} g_M(\{p, q\}) = 0$.

Let $n \in \mathbb{N}$ be arbitrary. Now we can compute the $n+2$ -nd bit of the dyadic representation of $g_M(\{p, q\})$. Due to the channels, we obviously have $g_M(\{p, q\}) \leq 1$. Let $M_0 = M_0(n)$ the smallest natural number such that the first $n+2$ bits of the dyadic representation of $g_{M_0}(\{p, q\})$ are zero. Then it holds for all $M \geq M_0$ that the $n+2$ -nd bit of the dyadic representation of $g_M(\{p, q\})$ is zero as well due to the monotonic convergence. But this implies that $g_{M_0}(\{p, q\}) = \sum_{k=n+3}^{\infty} a_k \frac{1}{2^k}$, $a_k \in \{0, 1\}$ and therewith $g_{M_0}(\{p, q\}) \leq \sum_{k=n+3}^{\infty} \frac{1}{2^k} = \frac{1}{2^{n+3}} \sum_{k=0}^{\infty} \frac{1}{2^k} = \frac{1}{2^{n+2}}$. With this we obtain

$$0 \leq \overline{C}(M_0; \{p, q\}) - \underline{C}(M_0; \{p, q\}) < \frac{1}{2^{n+2}}.$$

For $M \geq M_0$ we have $\overline{C}(M; \{p, q\}) \leq \overline{C}(M_0; \{p, q\})$ and $\underline{C}(M; \{p, q\}) \geq \underline{C}(M_0; \{p, q\})$. Thus, for $M \geq M_0$ we obtain

$$\begin{aligned} 0 &\leq \overline{C}(M; \{p, q\}) - \underline{C}(M; \{p, q\}) \\ &\leq \overline{C}(M_0; \{p, q\}) - \underline{C}(M; \{p, q\}) \\ &\leq \overline{C}(M_0; \{p, q\}) - \underline{C}(M_0; \{p, q\}) \\ &< \frac{1}{2^{n+2}}. \end{aligned}$$

Due to $C(\{p, q, s_0\}) = \underline{C}(\{p, q\}) = \lim_{M \rightarrow \infty} \underline{C}(M; \{p, q\})$ for all $s_0 \in \mathcal{S}$, we further have

$$0 \leq \overline{C}(M; \{p, q\}) - C(\{p, q, s_0\}) < \frac{1}{2^{n+2}}.$$

The function $M_0 = M_0(n)$ is effectively computable, since it is sufficient to run our algorithm until $a_{n+2}(g_{M_0}(\{p, q\})) = 0$ is satisfied which completes the proof. \square

Remark 30. For every computable FSC $\{p, q, s_0\}$, $s_0 \in \mathcal{S}$, that satisfies $\overline{C}(\{p, q\}) = \underline{C}(\{p, q\})$, we have $C(\{p, q, s_0\}) \in \mathbb{R}_c$ for all $s_0 \in \mathcal{S}$, i.e., the capacity is a computable real number. This means that there exists an algorithm for the inputs p, q that computes the desired approximation of the number $C(\{p, q, s_0\})$. In general, this algorithm does not depend recursively on the input $\{p, q\}$. Theorem 19 actually shows that this dependency must be non-recursive in general, since $C(\{p, q, s_0\})$ is not even Banach-Mazur computable in $\{p, q, s_0\}$.

Remark 31. If there exist $\{\hat{p}, \hat{q}\} \in \mathcal{P}_c \times \mathcal{Q}_c$ and $s_0 \in \mathcal{S}$ such that $C(\{\hat{p}, \hat{q}, s_0\}) \notin \mathbb{R}_c$, then this is the strongest form of non-computability, since then the value $C(\{\hat{p}, \hat{q}, s_0\})$ is not algorithmically computable although the parameters $\{\hat{p}, \hat{q}\} \in \mathcal{P}_c \times \mathcal{Q}_c$ are computable real numbers. In [33, 37] it has been shown that there exist computable compound and averaged channels, where the state set is countably infinite, for which this phenomenon appears, i.e., there are computable compound and averaged channels such that its capacity is not a computable real number. This implies that for certain fixed computable compound or averaged channels, there exists no algorithm for the computation of the capacity. In [33] it has been further shown that such channels can be constructed based on binary symmetric channels. In addition to that, it has been shown that the achievability part, i.e., the coding part, cannot be constructive, i.e., there is no algorithm that can construct the corresponding encoder and decoder. This is particularly interesting to observe given the recent progress in polar codes that can construct algorithmically capacity-achieving encoder and decoder for fixed computable binary symmetric channels. The result in [33] on the other hand shows that this is no longer possible in general for compound and averaged channels.

Some further comments are in order:

- There are several definitions of computable functions which are not equivalent in general.
- The notion of Banach-Mazur computability is the weakest notion of computability.
- From a practical point of view, it is not clear if it makes sense to further weaken the requirements of Banach-Mazur computable functions.
- It is common sense that a computable function should map computable numbers from its domain to computable numbers within its value range.

To this end, it is interesting to observe that computable compound and averaged channels need not necessarily satisfy this basic requirement, cf. [33], where computable channels are constructed whose capacity is a non-computable real number.

5. FSC Capacity as an Optimization Problem

Let us go back one more time to Theorem 19 and its proof, where we analyzed the capacity function $C(\{p, q, s_0\}) : \mathcal{P}_c \times \mathcal{Q}_c \times \mathcal{S} \rightarrow \mathbb{R}$. We have shown that the capacity function $C(\{p, q, s_0\})$ is discontinuous for certain $s_0 \in \mathcal{S}$ and computable $p \in \mathcal{P}_c$ and $q \in \mathcal{Q}_c$.

Theorem 32. *For all $|\mathcal{X}| \geq 2$, $|\mathcal{Y}| \geq 2$, and $|\mathcal{S}| \geq 2$, the capacity function $C : \mathcal{P} \times \mathcal{Q} \times \mathcal{S} \rightarrow \mathbb{R}$ is discontinuous.*

Proof. We consider the channels $p(y_n|x_n, 0)$, $p(y_n|x_n, 1)$, $\hat{q}(y_n|x_n, 0)$, and $\hat{q}(y_n|x_n, 1)$ as in (11) and (12).

Next, we consider $\{p, q_k, s_0\}$ for $k \geq 1$ with

$$(19) \quad q_k(s_n|x_n, 0) = \left(\frac{1 - \frac{1}{k+1}}{\frac{1}{k+1}} \right), \quad q_k(s_n|x_n, 1) = \left(\frac{\frac{1}{k+1}}{1 - \frac{1}{k+1}} \right).$$

We observe that the FSC $\{p, q_k, s_0\}$, $s_0 \in \mathcal{S}$, $k \geq 1$, as defined above is indecomposable. Further, q_k is obviously computable, i.e., $q_k \in \mathcal{Q}_c$, and further independent of $x_n \in \mathcal{X}$.

Next, we need a concept of distance. For $p^{(1)}, p^{(2)} \in \mathcal{P}_c$ and $q^{(1)}, q^{(2)} \in \mathcal{Q}_c$ we define the distance between the FSCs $\{p^{(1)}, q^{(1)}, s_0\}$ and $\{p^{(2)}, q^{(2)}, s_0\}$ as

$$(20) \quad \begin{aligned} & d(\{p^{(1)}, q^{(1)}, s_0\}, \{p^{(2)}, q^{(2)}, s_0\}) \\ &= \max_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} |p^{(1)}(y|x, s_0) - p^{(2)}(y|x, s_0)| \\ & \quad + \max_{x \in \mathcal{X}} \sum_{s \in \mathcal{S}} |q^{(1)}(s|x, s_0) - q^{(2)}(s|x, s_0)|. \end{aligned}$$

For FSCs as defined in (11)-(19), we have for any $s_0 \in \mathcal{S}$, $d(\{p, \hat{q}, s_0\}, \{p, q_k, s_0\}) = \frac{2}{k+1}$.

Next, let us assume that $C(\{p, q, s_0\})$, $s_0 \in \{0, 1\}$, is Banach-Mazur computable on $\mathcal{P}_c \times \mathcal{Q}_c$. Then this would require that both capacities for $s_0 = 0$ and $s_0 = 1$ are continuous functions on $\mathcal{P}_c \times \mathcal{Q}_c$. In particular, we must have $\lim_{k \rightarrow \infty} C(\{p, q_k, 0\}) = C(\{p, q, 0\})$ and $\lim_{k \rightarrow \infty} C(\{p, q_k, 1\}) = C(\{p, q, 1\})$.

Since for all $k \in \mathbb{N}$ the FSC $\{p, q_k, s_0\}$, $s_0 \in \mathcal{S}$, is indecomposable, we have $C(\{p, q_k, 0\}) = C(\{p, q_k, 1\})$ and further obtain

$$\begin{aligned} 1 = C(\{p, q, 0\}) &= \lim_{k \rightarrow \infty} C(\{p, q_k, 0\}) = \lim_{k \rightarrow \infty} C(\{p, q_k, 1\}) \\ &= C(\{p, q, 1\}) = C_{\text{BSC}}(\epsilon) = 1 - H_2(\epsilon) < 1 \end{aligned}$$

which is a contradiction. Accordingly, at least one of the functions $C(\{p, q, 0\})$ or $C(\{p, q, 1\})$ must be discontinuous proving the desired result. \square

This allows to obtain the following result.

Theorem 33. *Let $|\mathcal{X}| \geq 2$, $|\mathcal{Y}| \geq 2$, and $|\mathcal{S}| \geq 2$ be arbitrary. Then there is no natural number $n_0 \in \mathbb{N}$ such that the capacity $C(\{p, q, s_0\})$ can be expressed as*

$$(21) \quad C(\{p, q, s_0\}) = \max_{u \in \mathcal{U}} F(u, p, q, s_0)$$

with $\mathcal{U} \subset \mathbb{R}^{n_0}$ a compact set and $F : \mathcal{U} \times \mathcal{P} \times \mathcal{Q} \times \mathcal{S} \rightarrow \mathbb{R}$ a continuous function.

Sketch of Proof. The result can be shown similarly as in [38]. The crucial observation is the following: To be able to express the capacity $C(\{p, q, s_0\})$ as in (21), the capacity necessarily needs to be a continuous function which cannot be the case by Corollary 32. \square

Remark 34. Theorem 33 further immediately implies that the capacity C cannot be expressed by a finite multi-letter formula. As a consequence, if C can be described by entropic quantities, then this must be done via a corresponding sequence. Accordingly, the characterization via a limit of multi-letter expressions cannot be simplified and there is no closed form solution possible in general for the capacity of FSCs.

6. Discussion and Open Problems

In this paper, we have studied the capacity of FSCs and we have shown that the capacity function $C(\{p, q, s_0\})$ is not Banach-Mazur computable. As a consequence, the capacity does not depend recursively on the system parameters $\{p, q, s_0\}$ and it is impossible to algorithmically compute the capacity $C(\{p, q, s_0\})$. We have further shown that we cannot find tight

lower and upper bounds on the capacity which are simultaneously computable continuous functions or Borel computable functions, respectively. This means that either the achievability or the converse (or both) must result in non-computable lower or upper bounds. It is not known which of them are actually non-computable for the FSC and, accordingly, the implications on the information theoretic approaches of the achievability and converse are unknown. Furthermore, the capacity is also shown to be non-approximable, i.e., it is impossible to approximate the capacity by computable sequences of computable functions for certain approximation errors.

For certain applications however, algorithmically computing the capacity of an FSC might be more than is actually needed. For example for resource allocation, it is often sufficient to know whether or not the current channel supports a certain quality-of-service (QoS) requirement λ . Accordingly, the following question is of interest:

Question 5: Is there an algorithm (or Turing machine) that takes the FSC $\{p, q, s_0\}$ and the QoS requirement $\lambda > 0$ as inputs and outputs “yes” if $C(\{p, q, s_0\}) > \lambda$ and outputs “no” if $C(\{p, q, s_0\}) < \lambda$?

This is a *decision* problem where the Turing machine decides whether or not an FSC supports a certain QoS requirement. Note that this Turing machine necessarily needs to stop for all possible inputs. However, it is not clear if problem is decidable and that such a Turing machine actually exists. In such a case, one may be inclined to weaken the question as follows:

Question 6: Is there an algorithm (or Turing machine) that takes the FSC $\{p, q, s_0\}$ and the QoS requirement $\lambda > 0$ as inputs and stops if $C(\{p, q, s_0\}) > \lambda$?

This modified question asks whether or not it is *semidecidable*. Here, the Turing machine must only stop and output the correct answer if the FSC supports the QoS requirement, i.e., $C(\{p, q, s_0\}) > \lambda$. In the other case, it does not stop and runs forever. It is clear that one can pose this question also in the opposite way by requiring the Turing machine to stop only if $C(\{p, q, s_0\}) < \lambda$.

There are several communication scenarios and channels whose capacity functions are not Turing computable, but their corresponding decision problems are semidecidable, cf. [39]. It is of interest to study such questions also for FSCs.

We want to conclude by coming back one more time to Kailath's work in information theory and the characterization of time-variant channels. In this case, the corresponding characterizations of capacities according to our results remain unknown. But as already mentioned in the introduction, there are further connections to the effective analysis and constructive mathematics. Here, the aim is to solve certain mathematical questions effectively, i.e., with the help of algorithms.

Recently, impressive progress has been made in the theory of time-variant channels. For a detailed discussion we refer to [40]. For example, progress in the design of test signals for channel identification [41, 42], extension to the multiple-input multiple-output (MIMO) case [43, 44], stochastic channels [45, 46], channels with unknown carrier [43, 47], constraints on the channel estimation [48], and others. These results address many of the problems discussed in [2] and provide solutions based on the classical analysis. In these works, methods such as distribution theory have been used that are not effective in general. This means that only the existence of certain strategies has been shown without the provision of effective algorithms or proofs. Note that this does not immediately exclude the possibility of a constructive characterization. But we want to note that in [49] computable absolutely integrable band-limited signals have been constructed, which are then also computable signals in $L^2(\mathbb{R})$, for which the bandwidth $B(f)$ is not a computable real number. It is not clear if this yields the impossibility of effective characterizations of the results in the above mentioned works.

Acknowledgment

Holger Boche would like to thank Volker Pohl for insightful discussions on time-continuous channels. He would like to further thank Robert Schober for interesting and fruitful discussions on the application of FSCs and time-continuous channels in molecular communication.

This work of H. Boche was supported in part by the German Federal Ministry of Education and Research (BMBF) within the national initiative for “*Molecular Communication (MAMOKO)*” under Grant 16KIS0914 and in part by the German Research Foundation (DFG) within the Gottfried Wilhelm Leibniz Prize under Grant BO 1734/20-1 and within Germany's Excellence Strategy – EXC-2111 – 390814868. This work of R. F. Schaefer was supported in part by the BMBF within the national initiative for “*Post Shannon Communication (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 Grants CCF-0939370, CCF-1513915, and CCF-1908308.

This paper was presented in part at the IEEE Information Theory Workshop (ITW), Visby, Sweden, Aug. 2019 [50] and in part at the National Research Meeting on Molecular Communications at the Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany, Dec. 2018.

References

- [1] R. G. Gallager, *Information Theory and Reliable Communication*. New York, NY, USA: John Wiley & Sons, Inc., 1968.
- [2] T. Kailath, “Sampling models for linear time-variant filters,” Massachusetts Institute of Technology, Research Laboratory of Electronics, Tech. Rep. 352, May 1959.
- [3] —, “Correlation detection of signals perturbed by a random channel,” *IRE Trans. Inf. Theory*, vol. 6, no. 3, pp. 361–366, Jun. 1960.
- [4] —, “Communication via randomly varying channels,” Thesis, Massachusetts Institute of Technology, 1961. [Online]. Available: <http://hdl.handle.net/1721.1/11319>
- [5] —, “Measurements on time-variant communication channels,” *IRE Trans. Inf. Theory*, vol. 8, no. 5, pp. 229–236, Sep. 1962.
- [6] —, “Time-variant communication channels,” *IEEE Trans. Inf. Theory*, vol. 9, no. 4, pp. 233–237, Oct. 1963.
- [7] T. Nakano, A. W. Eckford, and T. Haraguchi, *Molecular Communication*. Cambridge, UK: Cambridge University Press, 2013.
- [8] D. Blackwell, “Information theory,” in *Modern Mathematics for the Engineer: Second Series*, E. F. Beckenbach and M. R. Hestenes, Eds. New York: McGraw-Hill Book Company, 1961, pp. 183–193.
- [9] R. Ahlswede and A. H. Kaspi, “Optimal coding strategies for certain permuting channels,” *IEEE Trans. Inf. Theory*, vol. IT-33, no. 3, pp. 310–314, May 1987.
- [10] K. Kobayashi and H. Morita, “An input/output recursion for the trapdoor channel,” in *Proc. IEEE Int. Symp. Inf. Theory*, Lausanne, Switzerland, Jun. 2002, p. 423.

- [11] H. Permuter, P. Cuff, B. Van Roy, and T. Weissman, “Capacity of the trapdoor channel with feedback,” *IEEE Trans. Inf. Theory*, vol. 54, no. 7, pp. 3150–3165, Jul. 2008.
- [12] D. Blackwell, L. Breiman, and A. J. Thomasian, “Proof of Shannon’s transmission theorem for finite-state indecomposable channels,” *Ann. Math. Statist.*, vol. 29, no. 4, pp. 1209–1220, 1958.
- [13] A. J. Goldsmith and P. P. Varaiya, “Capacity, mutual information, and coding for finite-state Markov channels,” *IEEE Trans. Inf. Theory*, vol. 42, no. 3, pp. 868–886, May 1996.
- [14] A. Lapidoth and I. E. Telatar, “The compound channel capacity of a class of finite-state channels,” *IEEE Trans. Inf. Theory*, vol. 44, no. 3, pp. 973–983, May 1998.
- [15] Z. Aharoni, O. Sabag, and H. H. Permuter, “Computing the feedback capacity of finite state channels using reinforcement learning,” in *Proc. IEEE Int. Symp. Inf. Theory*, Paris, France, Jul. 2019, pp. 837–841.
- [16] S. Verdú and T. S. Han, “A general formula for channel capacity,” *IEEE Trans. Inf. Theory*, vol. 40, no. 4, pp. 1147–1157, Jul. 1994.
- [17] A. M. Turing, “On computable numbers, with an application to the Entscheidungsproblem,” *Proc. London Math. Soc.*, vol. 2, no. 42, pp. 230–265, 1936.
- [18] —, “On computable numbers, with an application to the Entscheidungsproblem. A correction,” *Proc. London Math. Soc.*, vol. 2, no. 43, pp. 544–546, 1937.
- [19] K. Weihrauch, *Computable Analysis - An Introduction*. Berlin, Heidelberg: Springer-Verlag, 2000.
- [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] 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.
- [25] M. B. Pour-El and J. I. Richards, *Computability in Analysis and Physics*. Cambridge: Cambridge University Press, 2017.
- [26] D. Elkouss and D. Pérez-García, “Memory effects can make the transmission capability of a communication channel uncomputable,” *Nature Communications*, vol. 9, no. 1, p. 1149, Mar. 2018.
- [27] M. Agarwal, “Non-existence of certain kind of finite-letter mutual information characterization for a class of time-invariant Markoff channels,” 2018, available online at <https://arxiv.org/abs/1804.05977>.
- [28] R. I. Soare, *Recursively Enumerable Sets and Degrees*. Berlin, Heidelberg: Springer-Verlag, 1987.
- [29] W. Rudin, *Real and Complex Analysis*, 3rd ed. Mcgraw-Hill Higher Education, 1987.
- [30] E. Specker, “Nicht konstruktiv beweisbare Sätze der Analysis,” *Journal of Symbolic Logic*, vol. 14, no. 3, pp. 145–158, Sep. 1949.
- [31] R. L. Dobrushin, “General formulation of Shannon’s main theorem in information theory,” *Amer. Math. Soc. Trans.*, vol. 33, pp. 323–438, 1963.
- [32] T. S. Han and S. Verdú, “Approximation theory of output statistics,” *IEEE Trans. Inf. Theory*, vol. 39, no. 3, pp. 752–772, May 1993.
- [33] H. Boche, R. F. Schaefer, and H. V. Poor, “Communication under channel uncertainty: An algorithmic perspective and effective construction,” *IEEE Trans. Signal Process.*, 2020, will appear.
- [34] 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.

- [35] H. Boche, R. F. Schaefer, and H. V. Poor, “Identification capacity of channels with feedback: Discontinuity behavior, super-activation, and Turing computability,” *IEEE Trans. Inf. Theory*, vol. 66, no. 10, pp. 6184–6199, Oct. 2020.
- [36] —, “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.
- [37] —, “Robust transmission over channels with channel uncertainty: An algorithmic perspective,” in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Process.*, Barcelona, Spain, May 2020, pp. 5230–5234.
- [38] —, “Identification capacity of correlation-assisted discrete memoryless channels: Analytical properties and representations,” in *Proc. IEEE Int. Symp. Inf. Theory*, Paris, France, Jul. 2019.
- [39] —, “Resource allocation for secure communication systems: Algorithmic solvability,” in *Proc. 11th IEEE Int. Workshop Inf. Forensics Security*, Delft, The Netherlands, Dec. 2019, pp. 1–6.
- [40] D. Walnut, G. E. Pfander, and T. Kailath, “Cornerstones of sampling of operator theory,” in *Excursions in Harmonic Analysis, Volume 4: The February Fourier Talks at the Norbert Wiener Center*, R. Balan, M. Bugué, J. J. Benedetto, W. Czaja, and K. A. Okoudjou, Eds. Cham: Birkhäuser, 2015, pp. 291–332.
- [41] J. Lawrence, G. E. Pfander, and D. Walnut, “Linear independence of gabor systems in finite dimensional vector spaces,” *J Fourier Anal. Appl.*, vol. 11, no. 6, pp. 715–726, Dec. 2005.
- [42] G. E. Pfander and D. Walnut, “Measurement of time-variant linear channels,” *IEEE Trans. Inf. Theory*, vol. 52, no. 11, pp. 4808–4820, Nov. 2006.
- [43] —, “Sampling and reconstruction of operators,” *IEEE Trans. Inf. Theory*, vol. 62, no. 1, pp. 435–458, Jan. 2016.
- [44] D. G. Lee, G. E. Pfander, and V. Pohl, “Sampling and reconstruction of multiple-input multiple-output channels,” *IEEE Trans. Signal Process.*, vol. 67, no. 4, pp. 961–976, Feb. 2019.
- [45] G. E. Pfander and P. Zheltov, “Identification of stochastic operators,” *Appl. Comput. Harmon. Anal.*, vol. 36, no. 2, pp. 256–279, Mar. 2014.

- [46] ———, “Sampling of stochastic operators,” *IEEE Trans. Inf. Theory*, vol. 60, no. 4, pp. 2359–2372, Apr. 2014.
- [47] R. Heckel and H. Bölcskei, “Identification of sparse linear operators,” *IEEE Trans. Inf. Theory*, vol. 59, no. 12, pp. 7985–8000, Dec. 2013.
- [48] D. G. Lee, G. E. Pfander, V. Pohl, and W. Zhou, “Identification of channels with single and multiple inputs and outputs under linear constraints,” *Linear Algebra Appl.*, vol. 581, pp. 435–470, Nov. 2019.
- [49] H. Boche and U. J. Mönich, “Effective approximation of bandlimited signals and their samples,” in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Process.*, Barcelona, Spain, May 2020, pp. 5590–5594.
- [50] H. Boche, R. F. Schaefer, and H. V. Poor, “On the structure of the capacity formula for general finite state channels with applications,” in *Proc. IEEE Inf. Theory Workshop*, Visby, Sweden, Aug. 2019, pp. 1–5.

INSTITUTE OF THEORETICAL INFORMATION TECHNOLOGY
TECHNISCHE UNIVERSITÄT MÜNCHEN
MUNICH, GERMANY
E-mail address: boche@tum.de

INFORMATION THEORY AND APPLICATIONS CHAIR
TECHNISCHE UNIVERSITÄT BERLIN
BERLIN, GERMANY
E-mail address: rafael.schaefer@tu-berlin.de

DEPARTMENT OF ELECTRICAL ENGINEERING
PRINCETON UNIVERSITY
PRINCETON, NJ 08544, USA
E-mail address: poor@princeton.edu