Discrete Mathematics and Theoretical Computer Science DMTCS vol. **10:3**, 2008, 75–94

Shifts with Decidable Language and Non-Computable Entropy

Peter Hertling[†] and Christoph Spandl

Institut für Theoretische Informatik und Mathematik, Fakultät für Informatik, Universität der Bundeswehr München, 85577 Neubiberg, Germany

received March 8, 2007, *revised October 2, 2008*, *accepted October 3, 2008*.

We consider subshifts of the full shift of all binary bi-infinite sequences. On the one hand, the topological entropy of any subshift with computably co-enumerable language is a right-computable real number between 0 and 1. We show that, on the other hand, any right-computable real number between 0 and 1, whether computable or not, is the entropy of some subshift with even polynomial time decidable language. In addition, we show that computability of the entropy of a subshift does not imply any kind of computability of the language of the subshift.

Keywords: Shift spaces, topological entropy, language of a shift space, computability, computable real numbers, right computable real numbers

1 Introduction

An important numerical quantity associated with a dynamical system is its topological entropy. A natural question is then: under what circumstances can one compute the topological entropy of a dynamical systems? While for specific types of dynamical systems, one can often give an algorithm, for examples see, e.g., Milnor (2002), there are also a number of negative results, e.g., for cellular automata by Hurd et al. (1992), for piecewise affine maps by Koiran (2001), and for shift spaces by Simonsen (2006) as well as by the second author, Spandl (2007). Shift spaces are an important, standard class of dynamical systems. For an introduction the reader is referred to Lind and Marcus (1995). They are closed, shift invariant subspaces of the space of bi-infinite sequences over some alphabet, usually assumed to be finite. For simplicity we will consider only subshifts of the space of all bi-infinite binary sequences. Whenever we speak about a *shift space* or simply a *shift* in this paper, we will always mean a closed, shift invariant subspace of the space of bi-infinite binary sequences.

As is well known, for simple types of shifts one can compute the entropy, for example for shifts that can be defined by a finite set of forbidden words, and also for the larger class of so-called sofic shifts; see Simonsen (2006); Spandl (2007). Furthermore, Simonsen (2005) obtained computability results for

[†]During the preparation of this work the first author was partially supported by DFG (grants no. HE 2489/4-1, 446 CHV 113/240/0-1), by NSFC (grant no. 10420130638), and by JSPS (grant no. 16340028).

the entropy of β-shifts, Spandl (2007), as well as Hertling and Spandl (2008), for the entropy of S-gap shifts, and Spandl (2007) also for the entropy of shifts having the specification property. Hochman and Meyerovitch (2007) gave a computability-theoretic characterization similar to our main result of the set of real numbers that are the entropy of a two-dimensional shift of finite type.

A natural way of describing a general shift is via its language, that is the set of all finite strings that appear as substrings of elements of the shift space. Simonsen (2006) and Spandl (2007) have shown that the entropy is not uniformly computable with respect to the language of the shift. To be precise, Simonsen (2006) has shown that there is no Turing machine which, given a decision algorithm for the language of some shift with decidable language, is able to compute the entropy of the shift with arbitrary precision. Spandl (2007) has shown that there is no oracle Turing machine which, given the language of some shift via an oracle, is able to compute the entropy of the shift with arbitrary precision. Simonsen (2006) posed the question whether there exists a shift with decidable language whose entropy is a non-computable number, and noted that "then the results in this paper related to uncomputability would follow immediately". In this paper, we construct such a shift. In fact we will show more. The second author, Spandl (2007), showed that, given an enumeration of the complement of the language of some shift space, one can approximate the entropy effectively from the right. Hence, if the complement of the language of a nonempty shift is computably enumerable then the entropy of the shift is a right-computable real number (between 0 and 1). In particular, the entropy of any nonempty shift with decidable language is a right-computable real number between 0 and 1. We will show that conversely any right-computable real number between 0 and 1 is the entropy of a shift with decidable language. In fact, it is even the entropy of an irreducible shift with a polynomial time decidable language. Since the class of right-computable real numbers between 0 and 1 is known to be strictly larger than the class of computable real numbers between 0 and 1, this shows that there exist irreducible shifts with polynomial time decidable language that have non-computable entropy.

In the following section, we give some basic definitions about shifts, forbidden words, and the language of a shift. Note that another natural way of representing a shift space is by giving a so-called set of forbidden words for it. In view of the question whether, given some information about a shift, one can compute its entropy, it is of course interesting to have a clear picture of the relation between different kinds of information about a shift. Here, we analyze the relation between two different kinds of information about a general shift, namely: language or set of forbidden words, and strengthen the observation by Simonsen (2006) that knowing the language of a shift is better than knowing only a set of forbidden words for the shift. Then we define the topological entropy. We observe that the obstacle to computing the topological entropy of a shift from its language is a topological one. Then we state the main result of the paper. We finish the section by considering the inverse question: if the topological entropy of a shift is computable, does this imply that the language is decidable, at least if the complement of the language is known to be c.e.? We show that this is not the case. On the contrary, we show that there exist shifts with topological entropy zero whose language is of any desired degree of computability-theoretic complexity. In the penultimate section, we prove the main result. We conclude with a summary and some open problems.

2 Shifts, Sets of Forbidden Words, and Languages of Shifts

We write $\mathbb{N} = \{0, 1, 2, \ldots\}$ for the set of natural numbers, i.e., of non-negative integers, $\Sigma = \{0, 1\}$ for the *binary alphabet*, Σ^* for the set of all finite binary strings, λ for the empty string, Σ^n for the set of all binary strings of length n, for any $n \in \mathbb{N}$. A subset of Σ^* is called (binary) *language*. A subset of Σ^* or of $\mathbb{N}$ is called *computably co-enumerable* (*co-c.e.*) if its complement is computably enumerable (*c.e.*). Furthermore, we write $\Sigma^\omega := \{p \mid p : \mathbb{N} \to \Sigma\}$ for the set of all one-way infinite binary sequences, and $\Sigma^\mathbb{Z} := \{p \mid p : \mathbb{Z} \to \Sigma\}$ for the set of all bi-infinite binary sequences. For $p \in \Sigma^\mathbb{Z}$ and $n \in \mathbb{Z}$, we write $p[n]$ for the value of p on input n, and $p[m \dots n] := p[m] \dots p[n]$, if $m \leq n$, and $p[m \dots n] := \lambda$, if $m > n$. For $w \in \Sigma^*$ we set $w\Sigma^\omega := \{p \in \Sigma^\omega \mid w \text{ is a prefix of } p\}$ and denote by $|w|$ the length of w. A string $w \in \Sigma^*$ is a *substring* of some $p \in \Sigma^\mathbb{Z}$ if there is an $n \in \mathbb{Z}$ with $w = p[n+1 \dots n+|w|]$. For any sets X and Y, we write $f :\subseteq X \to Y$ for a function whose domain $\mathrm{dom}(f)$ is a subset of X and whose range is a subset of Y. In case $\mathrm{dom}(f) = X$ we may write $f : X \to Y$. Endowed with the product topology of the discrete topology on Σ, both Σ^ω and $\Sigma^\mathbb{Z}$ are compact topological spaces. The function $\sigma : \Sigma^\mathbb{Z} \to \Sigma^\mathbb{Z}$ defined by

$$\sigma(p)[n] := p[n+1] \quad \text{for } p \in \Sigma^\mathbb{Z} \text{ and } n \in \mathbb{Z}$$

is called the *shift map*. It is a homeomorphism.

Definition 1 We call a subset $X \subseteq \Sigma^\mathbb{Z}$ a *shift* or a *shift space* or a *subshift* of $\Sigma^\mathbb{Z}$ if it is closed and satisfies $\sigma(X) = X$.

Examples of shift spaces are easily obtained.

Lemma 2 *For an arbitrary set $F \subseteq \Sigma^*$ the set $X_F \subseteq \Sigma^\mathbb{Z}$ defined by*

$$X_F := \{p \in \Sigma^\mathbb{Z} \mid \textit{no string in } F \textit{ is a substring of } p\}$$

is a shift space.

Proof: See (Lind and Marcus, 1995, Definition 1.2.1 and Theorem 6.1.21). □

Definition 3 If $F \subseteq \Sigma^*$ is a set with $X = X_F$, then F is called a *set of forbidden words for X*.

This poses the question whether every shift space can be defined via a set of forbidden words, i.e., whether for every shift space X there is a set F with $X = X_F$. The answer is yes, as can be seen via the "language" of the shift space.

Definition 4 For any set $X \subseteq \Sigma^\mathbb{Z}$ the set

$$\mathcal{L}(X) := \{w \in \Sigma^* \mid w \text{ is a substring of some } p \in X\}$$

is the *language* of X.

For any shift space X, the complement $\mathcal{L}(X)^c$ of $\mathcal{L}(X)$ is a set of forbidden words for X, i.e., $X = X_{\mathcal{L}(X)^c}$ (indeed, $X \subseteq X_{\mathcal{L}(X)^c}$ is clear, and $X_{\mathcal{L}(X)^c} \subseteq X$ follows from the assumption that X is closed). In fact, it is the largest forbidden set of words for X since any other set of forbidden words for X is obviously a subset of $\mathcal{L}(X)^c$.

Example 5 For the shift space $X := \{0^\mathbb{Z}\}$ containing only the bi-infinite sequence containing only 0's, one sees $\mathcal{L}(X) = \{0\}^*$, hence, $\mathcal{L}(X)^c = \{0,1\}^* \setminus \{0\}^*$. Any set F with $\{1\} \subseteq F \subseteq \{0,1\}^* \setminus \{0\}^*$ is a set of forbidden words for X. Also the sets $\{10, 11\}$ and $\{01, 11\}$ and so on are sets of forbidden words for X.

A particularly simple class of shifts are those that possess a finite set of forbidden words. But in general, a shift does not need to have a finite set of forbidden words. So, if one wishes to compute something associated with a shift, e.g., its entropy, one needs some kind of description of the shift as input for the algorithm. What is a natural description of a general shift X? One might assume that one knows the characteristic sequence of either a set of forbidden words for X or of $\mathcal{L}(X)$. Here the *characteristic sequence* χ_S of a set $S \subseteq \Sigma^*$ is the binary sequence $\chi_S \in \Sigma^\omega$ defined by

$$\chi_S(n) := \begin{cases} 1 & \text{if } \nu(n) \in S, \\ 0 & \text{if } \nu(n) \notin S \end{cases}$$

where the bijection $\nu : \mathbb{N} \to \Sigma^*$ is defined by the length-lexicographical ordering: order all strings in Σ^* according to their length (short strings first), and strings of the same length in alphabetical order.

We have just seen that, given the language of a shift, one can easily compute a set of forbidden words for the shift: just take the complement of the language of the shift. But (Simonsen, 2006, Remark 17) showed that there is no Turing machine which, given a decision algorithm for some decidable set $F \subseteq \Sigma^*$, computes a decision algorithm for the language $\mathcal{L}(X_F)$. The argument used by Simonsen in order to prove this statement is actually a topological one: the function mapping the characteristic sequence of an arbitrary forbidden set of a shift to the characteristic sequence of its language is not continuous and can therefore not be computable. Note that any function $H :\subseteq \Sigma^\omega \to \Sigma^\omega$ that can be computed by a (Type-2) Turing machine is continuous; see, e.g., Weihrauch (2000).

Proposition 6 *The function $G : \Sigma^\omega \to \Sigma^\omega$, defined by $G(\chi_F) := \chi_{\mathcal{L}(X_F)}$ is not continuous.*

Proof: A point of discontinuity is for example the sequence $0^\omega = \chi_\emptyset$. Note that $G(0^\omega) = 1^\omega$ because $\mathcal{L}(X_\emptyset) = \mathcal{L}(\Sigma^\mathbb{Z})$ is the whole set Σ^*. If G were continuous in 0^ω, there should be a finite prefix 0^n of 0^ω such that $G(0^n\Sigma^\omega) \subseteq 1\Sigma^\omega$. But if F is the set of all binary strings of length greater than $\log_2(n)$ then χ_F also starts with 0^n, but $X_F = \emptyset$, hence $\mathcal{L}(X) = \emptyset$, hence $\chi_{\mathcal{L}(X)} = 0^\omega$. □

We will strengthen Simonsen's remark (Simonsen, 2006, Remark 17) in the following way: in Corollary 11 we will see that there exists a decidable set F such that $\mathcal{L}(X_F)$ is not decidable. That means that there exists computable input for the function G in Proposition 6 which is mapped by this function to non-computable output. Note that any computable function $H :\subseteq \Sigma^\omega \to \Sigma^\omega$ maps any computable binary sequence $p \in \operatorname{dom} H$ to a computable binary sequence; see, e.g., Weihrauch (2000). Before we construct such a computable input, let us clarify what one can compute when one knows (an oracle for) some set F, i.e., if χ_F for some set F is given. We will see that one can still enumerate $\mathcal{L}(X_F)^c$. In fact in order to be able to do that, it is sufficient to be able to enumerate F. We formulate only a non-uniform (not involving oracle Turing machines) version of this statement.

Proposition 7 *If $F \subseteq \Sigma^*$ is a computably enumerable set, then $\mathcal{L}(X_F)^c$ is computably enumerable as well.*

Proof: If a set F is computably enumerable, then also the set of all strings in Σ^* that have a substring in F is computably enumerable. The assertion follows now from the following lemma. □

Lemma 8 *Fix some set $F \subseteq \Sigma^*$. A string v is in $\mathcal{L}(X_F)^c$ if, and only if, there exists some $n \in \mathbb{N}$ such that each of the 2^{2n} strings uvw with $u, w \in \Sigma^n$ has a substring in F.*

Proof: For a string v we set $K_v := \{p \in \Sigma^{\mathbb{Z}} \mid v = p[0 \dots |v|-1]\}$. A string v is in $\mathcal{L}(X_F)^c$ if, and only if, every $p \in K_v$ contains an element of F as a substring. And this is the case if, and only if, for any $p \in K_v$ there is some $n \in \mathbb{N}$ such that $p[-n \dots |v|-1+n]$ contains an element of F as a substring. Now, the claim follows easily from the compactness of the set K_v, for any v. □

Especially, if a shift X has a decidable set of forbidden words, then $\mathcal{L}(X)^c$ must be a c.e. set. The converse is true as well.

Theorem 9 *If X is a shift such that $\mathcal{L}(X)^c$ is c.e., then there exists a polynomial time decidable set $F \subseteq \Sigma^*$ with $X = X_F$.*

Proof: This is proved by a padding argument. Let X be a shift such that $\mathcal{L}(X)^c$ is computably enumerable. Let M be a Turing machine that halts on input $v \in \Sigma^*$ after finitely many steps if, and only if, $v \in \mathcal{L}(X)^c$. Let $h :\subseteq \Sigma^* \to \mathbb{N}$ be defined by

$$h(v) := \begin{cases} \text{number of steps until } M \text{ halts on input } v & \text{if } v \in \mathcal{L}(X)^c, \\ \text{undefined} & \text{if } v \in \mathcal{L}(X). \end{cases}$$

We define a set $F \subseteq \Sigma^*$ as follows:

$$F := \bigcup_{v \in \mathcal{L}(X)^c} \{vw \mid w \in \{0,1\}^{h(v)}\}.$$

We claim that this set F has the desired properties.

"$X = X_F$": This is clear.

"F is decidable in polynomial time": A Turing machine that decides for a given string $x \in \Sigma^*$ in time polynomial in $|x|$ whether x belongs to F or not can work for example as follows. For every prefix v of x it starts M with input v, lets M run for at most $|x| - |v| + 1$ steps and accepts x if, and only if, there is a prefix v of x such that M stops on input v after exactly $|x| - |v|$ steps. □

Corollary 10 *For a shift X the following conditions are equivalent:*

1. *$\mathcal{L}(X)^c$ is computably enumerable.*
2. *X has a c.e. set of forbidden words.*
3. *X has a polynomial time decidable set of forbidden words.*

Proof: "3 ⇒ 2": This is trivial.

"2 ⇒ 1": This is Proposition 7.

"1 ⇒ 3": This is Theorem 9. □

Now we easily conclude that there is computable input for the function G in Proposition 6 which is mapped by G to non-computable output.

Corollary 11 *There exists a polynomial time decidable set $F \subseteq \Sigma^*$ such that $\mathcal{L}(X_F)$ is not decidable.*

Proof: In order to derive this from Theorem 9 it is sufficient to observe that there exist shifts whose language is co-c.e., but not decidable. For an infinite set $S \subseteq \mathbb{N}$, define

$$F(S) := \{10^n 1 \mid n \in \mathbb{N} \setminus S\}.$$

Fix some co-c.e., but not decidable set $S \subseteq \mathbb{N}$ (such a set must be infinite). Then $\mathbb{N} \setminus S$ and $F(S)$ are c.e. and not decidable. By Proposition 7 also $\mathcal{L}(X_{F(S)})^c$ is computably enumerable. But $\mathcal{L}(X_{F(S)})^c$ is not decidable because a string of the form $10^n 1$ is in $\mathcal{L}(X_{F(S)})^c$ if, and only if, $n \in \mathbb{N} \setminus S$. □

Remark 12 For an infinite set $S \subseteq \mathbb{N}$, the shift $X_{F(S)}$ defined in the proof is called *S-gap shift* (Lind and Marcus, 1995, Example 1.2.6). For a finite, nonempty set $S \subseteq \mathbb{N}$, the shift $X_{F(S)}$ is called *S-gap shift* where

$$F(S) := \{10^n 1 \mid n \in \mathbb{N} \setminus S\} \cup \{0^{1+\max(S)}\};$$

see (Lind and Marcus, 1995, Example 1.2.6).

Corollary 11 strengthens Simonsen's observation (Simonsen, 2006, Remark 17) that there can be no Turing machine which, given a decision algorithm for any decidable set $F \subseteq \Sigma^*$, computes a decision algorithm for $\mathcal{L}(X_F)$.

3 The Entropy of Shifts with Decidable or with Computably Co-Enumerable Language

The following statement is Proposition 4.1.8 in Lind and Marcus (1995).

Proposition 13 *If X is a shift, then*

$$\lim_{n\to\infty} \frac{\log_2 |\mathcal{L}(X) \cap \Sigma^n|}{n}$$

exists, and equals

$$\inf_{n\geq 1} \frac{\log_2 |\mathcal{L}(X) \cap \Sigma^n|}{n}$$

Here we use $\log_2(0) := -\infty$.

Definition 14 For a shift X, the value $h_{\mathrm{top}}(X) := \lim_{n\to\infty} \frac{\log_2 |\mathcal{L}(X)\cap\Sigma^n|}{n}$ is called the *(topological) entropy* of X.

The entropy of the empty shift is $-\infty$. In the rest of the paper we will only consider nonempty shifts. It is clear that the entropy of a nonempty subshift of $\{0,1\}^{\mathbb{Z}}$ is always a real number between 0 and 1, i.e., a real number in the closed interval $[0,1]$. The converse is true as well; one can even restrict oneself to irreducible shifts.

Definition 15 A shift X is called *irreducible* if for any two strings $u, w \in \mathcal{L}(X)$ there is a string $v \in \Sigma^*$ with $uvw \in \mathcal{L}(X)$.

The irreducible shifts form an important subclass of shifts; compare Lind and Marcus (1995).

Proposition 16 *1. The entropy of a nonempty shift is a real number in the interval* $[0,1]$.

2. Any real number in $[0,1]$ *is the entropy of a nonempty, irreducible shift.*

Proof: The first assertion is clear. The second assertion follows immediately from (Lind and Marcus, 1995, Exercise 4.3.7(d)) where it is stated that every real number in $[0,1]$ is the entropy of an S-gap shift, as defined in Remark 12, following (Lind and Marcus, 1995, Example 1.2.6). It is clear that any S-gap shift is irreducible. □

A different proof of the second statement will be given in the following section. In fact, the proof of our main result (see below) will show how, given a non-increasing sequence of rational numbers in $[0,1]$, one can construct an irreducible shift whose entropy equals the limit of this sequence.

Can one compute the entropy of a shift? What kind of information does one need to have about the shift in order to compute its entropy? By (Spandl, 2007, Theorem 6.3) one can effectively approximate the entropy from the right if one has an enumeration of $\mathcal{L}(X)^c$. We state a non-uniform version of this precisely. Therefore, we need the notion of a right-computable real number. We directly define also the (more basic) notion of a computable real number.

Definition 17 1. A sequence $(q_n)_{n\in\mathbb{N}}$ of rational numbers is *computable* if there exist three total computable functions $s,a,b:\mathbb{N}\to\mathbb{N}$ with $b(n)\neq 0$ and $q_n=(-1)^{s(n)}\cdot a(n)/b(n)$, for all n.

2. A real number x is *computable* if there exists a computable sequence $(q_n)_{n\in\mathbb{N}}$ of rational numbers with $|x-q_n|\leq 2^{-n}$, for all $n\in\mathbb{N}$.

3. A real number x is *right-computable* if there exists a computable sequence $(q_n)_{n\in\mathbb{N}}$ of rational numbers with $q_n\geq q_{n+1}$ for all $n\in\mathbb{N}$ (a sequence satisfying this condition will be called *non-increasing*) and $x=\lim_{n\to\infty}q_n$.

Remark 18 Similarly, a real number is called *left-computable* if it is the limit of a computable non-decreasing sequence of rational numbers. Obviously, a real number x is right-computable if, and only if, $-x$ is left-computable. And a real number is computable if, and only if, it is left-computable and right-computable. But it is well known that the class of left-computable real numbers is strictly larger than the class of computable real numbers; the same applies to the class of right-computable real numbers. If, e.g., $K\subseteq\mathbb{N}$ is a c.e., but undecidable set, then the number $1-\sum_{n\in K}2^{-(n+1)}$ is a right-computable but not computable real number between 0 and 1; see, e.g., Weihrauch (2000). For an overview concerning left-computable real numbers and other classes of real numbers defined by computability properties, see Zheng (2002).

Proposition 19 *If X is a nonempty shift such that $\mathcal{L}(X)^c$ is c.e., then the entropy of X is a right-computable real number.*

Proof: This follows from $h_{\mathrm{top}}(X)=\inf_{n\geq 1}\frac{\log_2|\mathcal{L}(X)\cap\Sigma^n|}{n}$ (Proposition 13). It is also a direct consequence of (Spandl, 2007, Theorem 6.3). □

Corollary 20 *If a nonempty shift X has a c.e. set of forbidden words, then its entropy is a right-computable real number.*

Proof: By the previous proposition and Proposition 7. □

Of course, one would like not only to approximate the entropy from the right, but to compute it with any precision.

Simonsen (2006) and the second author, Spandl (2007), have shown that the entropy is not uniformly computable with respect to the language of the shift. To be precise, Simonsen (2006) has shown that there is no Turing machine which, given a decision algorithm for the language of some shift with decidable language, is able to compute the entropy of the shift with arbitrary precision. In fact, Simonsen (2006) showed that this negative result is true even if one restricts oneself to the class of irreducible shifts. The second author, Spandl (2007), showed that there is no oracle Turing machine which, given the language of some shift via an oracle, is able to compute the entropy of the shift with arbitrary precision. In fact, as was noted by Spandl (2007) and by Simonsen (2006) as well, these negative statements are true for topological reasons, similarly as the negative statement of Proposition 6 in the previous section was true for topological reasons.

Proposition 21 *The function $H :\subseteq \Sigma^\omega \to \mathbb{R}$ with*

$$\begin{aligned} \operatorname{dom} H &= \{\chi_{\mathcal{L}(X)} \mid X \text{ is a nonempty, irreducible shift}\}, \\ H(\chi_{\mathcal{L}(X)}) &= h_{\mathrm{top}}(X) \text{ for any such shift } X. \end{aligned}$$

is discontinuous. It is even discontinuous if on $\mathbb{R}$ the weaker topology is considered whose open sets are the sets (a, ∞), with $a \in \mathbb{R}$, and the sets $\emptyset$, $\mathbb{R}$.

The proof will be given in the following section. The proposition says essentially that there is no algorithm, in fact not even a continuous function, which, given the characteristic function of the language of a shift, would deliver arbitrarily good lower bounds to the entropy of the shift.

Simonsen (Simonsen, 2006, Page 94) then asked the question whether there exists a shift with decidable language but non-computable entropy. We shall show that this is indeed the case. Note that this strengthens the negative results by Simonsen (2006) and Spandl (2007) about the noncomputability of the entropy when the language of a shift is known. In fact, we shall show more. We have already seen that the entropy of any nonempty shift X with decidable language (even with co-c.e. language) is a right-computable real number between 0 and 1. We shall show that any such number is the entropy of a shift with decidable language. It is even the entropy of an irreducible shift with a polynomial time decidable language. The following theorem is the main result of this paper.

Theorem 22 *For every right-computable real number $\alpha \in [0, 1]$ there exists a nonempty, irreducible shift X with $h_{\mathrm{top}}(X) = \alpha$ and with polynomial time decidable language $\mathcal{L}(X)$.*

The proof will be given in the following section.

Corollary 23 *For a real number α the following conditions are equivalent.*

1. *$\alpha \in [0, 1]$ and α is right-computable.*
2. *There exists a shift X with co-c.e. language $\mathcal{L}(X)$ and entropy α.*
3. *There exists an irreducible shift X with polynomial time decidable language $\mathcal{L}(X)$ and entropy α.*

Proof: "3 ⇒ 2": Trivial.
"2 ⇒ 1": By Proposition 19 and Proposition 16.1.
"1 ⇒ 3": By Theorem 22. □

Corollary 24 *There exists an irreducible shift X with polynomial time decidable language $\mathcal{L}(X)$ such that its entropy $h_{\mathrm{top}}(X)$ is not a computable real number.*

Proof: In Remark 18 we already mentioned that there exist right-computable real numbers between 0 and 1 that are not computable. □

Remark 25 Our main result, Theorem 22, can be considered as an effective version of Proposition 16.2. We will see that by simplifying the proof (by forgetting about effectivity properties) one obtains a second proof of Proposition 16.2.

One may now also ask the other way around whether it might not be possible to prove our main result by effectivizing the proof given above for Proposition 16.2, i.e., by effectivizing the statement of (Lind and Marcus, 1995, Exercise 4.3.7(d)) that any real number in $[0, 1]$ is the entropy of an S-gap shift (compare Remark 12). Thus, the question arises whether for an arbitrary right-computable real number $\alpha \in [0, 1]$ there is an S-gap shift with entropy α and decidable language. The answer to this question is no, as was observed already by the second author in (Spandl, 2007, Section 7): for a nonempty set $S \subseteq \mathbb{N}$, the language of the S-gap shift $X_{F(S)}$ is decidable if, and only if, S is decidable, and in that case the entropy $h_{\mathrm{top}}(X_{F(S)})$ is a computable real number.

We have seen that decidability of the language of a shift implies right-computability of the entropy, but not computability of the entropy. One might ask also the inverse question: does computability of the entropy of a shift imply any computability property of the language of the shift, e.g., that is is decidable, perhaps at least if the language is already known to be co-c.e.? The next result shows that this is not the case. In order to formulate it we remind the reader of some notions from computability theory. A language $L \subseteq \Sigma^*$ is *many-one reducible* to a set $S \subseteq \mathbb{N}$ if there is a total computable function $h : \Sigma^* \to \mathbb{N}$ with $x \in L \iff h(x) \in S$, for any $x \in \Sigma^*$. Conversely, a set $S \subseteq \mathbb{N}$ is *many-one reducible* to $L \subseteq \Sigma^*$ if there is a total computable function $g : \mathbb{N} \to \Sigma^*$ with $x \in S \iff g(x) \in L$, for any $x \in \mathbb{N}$. Finally, $L \subseteq \Sigma^*$ and $S \subseteq \mathbb{N}$ are called *many-one equivalent* to each other if L is many-one reducible to S and S is many-one reducible to L. If two sets are many-one equivalent to each other then they have the same computability-theoretic degree of difficulty. For example, if one of them is decidable resp. c.e. resp. co-c.e., then the other one has the same property.

Theorem 26 *For any set $S \subseteq \mathbb{N}$ with $S \neq \emptyset$ and $S \neq \mathbb{N}$ there exists a shift over the binary alphabet whose topological entropy is equal to zero and whose language is many-one equivalent to S.*

Proof: For $s \in \mathbb{N}$ we define

$$X_{\mathrm{per}}(s) := \{c \in \{0,1\}^{\mathbb{Z}} \mid \begin{array}{l} c \text{ contains infinitely many 1's, and,} \\ \text{for any } k \in \mathbb{N}, \text{ the string } 10^k1 \text{ is a substring of } c \text{ if, and only if, } k = s\}. \end{array}$$

In other words, $X_{\mathrm{per}}(s)$ contains exactly the $s + 1$ bi-infinite binary periodic sequences of the form

$$\ldots 10^s10^s10^s1 \ldots .$$

For finite $S \subseteq \mathbb{N}$ with $S \neq \emptyset$, we define

$$X_{\text{per}}(S) := \bigcup_{s \in S} X_{\text{per}}(s),$$

and for infinite $S \subseteq \mathbb{N}$, we define

$$X_{\text{per}}(S) := \{0^{\mathbb{Z}}\} \cup \{p \in \Sigma^{\mathbb{Z}} \mid p \text{ contains exactly one } 1\} \cup \bigcup_{s \in S} X_{\text{per}}(s).$$

Let us fix a set $S \subseteq \mathbb{N}$ with $S \neq \emptyset$.

First, we show that $X_{\text{per}}(S)$ is a shift space. The set $X_{\text{per}}(S)$ is obviously invariant under the shift map σ, i.e., $\sigma(X_{\text{per}}(S)) = X_{\text{per}}(S)$. Furthermore, it is closed. In the case of infinite S, we ensured that $X_{\text{per}}(S)$ is closed by adding the sequence $0^{\mathbb{Z}}$ and the sequences in $\Sigma^{\mathbb{Z}}$ containing exactly one 1.

Now, we show that $h_{\text{top}}(X_{\text{per}}(S)) = 0$. Since $X_{\text{per}}(S)$ is nonempty and a subset of $X_{\text{per}}(\mathbb{N})$, it is sufficient to show that $h_{\text{top}}(X_{\text{per}}(\mathbb{N})) = 0$. Consider some $n > 0$. Let us give an upper bound on the number of strings in $\mathcal{L}(X_{\text{per}}(\mathbb{N})) \cap \Sigma^n$. Clearly,

$$\mathcal{L}(X_{\text{per}}(\mathbb{N})) \cap \Sigma^n = \{0^n\} \cup \bigcup_{s=0}^{n-1} (\mathcal{L}(X_{\text{per}}(s)) \cap \Sigma^n)$$

and, for any $s \in \mathbb{N}$,

$$|\mathcal{L}(X_{\text{per}}(s)) \cap \Sigma^n| \leq s + 1,$$

thus,

$$|\mathcal{L}(X_{\text{per}}(\mathbb{N})) \cap \Sigma^n| \leq 1 + \frac{n(n+1)}{2},$$

hence,

$$0 \leq h_{\text{top}}(X_{\text{per}}(\mathbb{N})) \leq \lim_{n \to \infty} \frac{\log_2(1 + n(n+1)/2)}{n} = 0.$$

In the rest of the proof we assume not only $S \neq \emptyset$, but also $S \neq \mathbb{N}$. We still have to show that $\mathcal{L}(X_{\text{per}}(S))$ is many-one equivalent to S. The set S is many-one reducible to $\mathcal{L}(X_{\text{per}}(S))$ because, for $s \in \mathbb{N}$,

$$s \in S \iff 10^s 1 \in \mathcal{L}(X_{\text{per}}(S)).$$

For the inverse claim (that $\mathcal{L}(X_{\text{per}}(S))$ is many-one reducible to S) we distinguish two cases.

1. S is finite. Then $\mathcal{L}(X_{\text{per}}(S))$ is obviously decidable. Thus, both sets are decidable. Since furthermore $\emptyset \subsetneqq S \subsetneqq \mathbb{N}$ and $\emptyset \subsetneqq \mathcal{L}(X_{\text{per}}(S)) \subsetneqq \Sigma^*$, they are many-one equivalent.

2. S is infinite. Then $\mathcal{L}(X_{\text{per}}(S))$ contains all strings that contain at most one 1. And a string w with two or more 1's is in $\mathcal{L}(X_{\text{per}}(S))$ if, and only if, it satisfies the following two conditions:

   (A) w contains at least two 1's, and there is some $s \leq |w|$ such that w is a substring of the periodic bi-infinite sequence

$$\ldots 10^s 10^s 10^s 1 \ldots,$$

(B) this number s (if is exists, it is obviously uniquely determined) is an element of S.

Condition (A) can be checked directly, and Condition (B) can be checked by a many-one reduction to S. To complete the proof formally, we define a computable reduction function $h_{\mathcal{L}(X_{\text{per}}(S)),S} : \Sigma^* \to \mathbb{N}$ from $\mathcal{L}(X_{\text{per}}(S))$ to S. Fix some $s_{\text{yes}} \in S$ and some $s_{\text{no}} \notin S$ (remember our assumptions $S \neq \emptyset$ and $S \neq \mathbb{N}$). We define

$$h_{\mathcal{L}(X_{\text{per}}(S)),S}(w) := \begin{cases} s_{\text{yes}} & \text{if } w \text{ contains at most one } 1, \\ s_{\text{no}} & \text{if } w \text{ contains at least two } 1\text{'s, but does not satisfy Condition (A),} \\ s & \text{if } w \text{ satisfies Condition (A), and } s \text{ is the number of } 0\text{'s between} \\ & \text{ any two } 1\text{'s in } w \text{ between which there is no other } 1. \end{cases}$$

It is clear that this function is computable and reduces $\mathcal{L}(X_{\text{per}}(S))$ to S.

We have shown that $X_{\text{per}}(S)$ has all the desired properties. □

Corollary 27 *There exists a shift over the binary alphabet whose entropy is zero and whose language is co-c.e., but not decidable.*

Proof: By taking for S in Theorem 26 any co-c.e., but undecidable set, one obtains such a shift. □

4 Proofs

In this section, we prove Proposition 21 and Theorem 22. In addition, we give a new proof of Proposition 16.2. Certain *sofic* shifts (compare Lind and Marcus (1995)) will play an important role.

Definition 28 1. A *finite* Σ*-labeled graph* is a pair (V, E) consisting of a finite set V (the set of *nodes*) and a finite set $E \subseteq V \times V \times \Sigma$ (the set of *edges labeled with elements of* Σ).

2. For a finite Σ-labeled graph (V, E), we define three functions $\text{start} : E \to V$, $\text{end} : E \to V$, and $\text{label} : E \to \Sigma$ giving the three components of a labeled edge as follows:

$$e = (\text{start}(e), \text{end}(e), \text{label}(e)),$$

for any $e \in E$.

3. A *bi-infinite path* through a finite Σ-labeled graph (V, E) is a function $q : \mathbb{Z} \to E$ such that $\text{end}(q(i)) = \text{start}(q(i + 1))$, for all $i \in \mathbb{Z}$.

Lemma 29 *If* (V, E) *is a finite* Σ*-labeled graph, then the following set is a shift and called the* sofic shift defined by the graph*:* $\{p \in \Sigma^{\mathbb{Z}} \mid$ *there exists a bi-infinite path* $q : \mathbb{Z} \to E$ *through the graph such that* $p[i] = \text{label}(q(i))$*, for all* $i \in \mathbb{N}\}$.

For the proof see (Lind and Marcus, 1995, Section 3.1).

By $\{0, *\}^+$ we denote the set of all nonempty strings over the alphabet $\{0, *\}$.

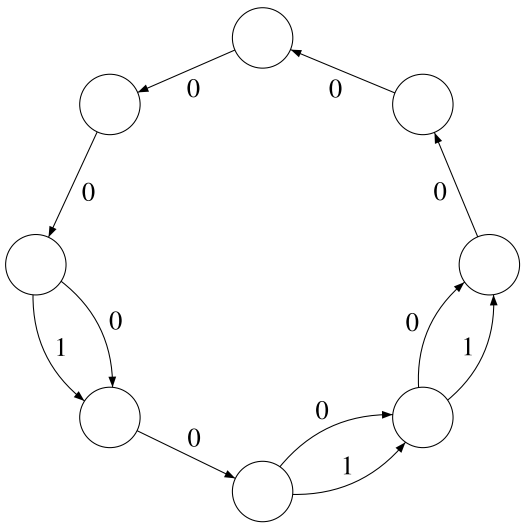

Fig. 1: Graph for the sofic shift $X(00*0**00)$.

Definition 30 For any $w \in \{0,*\}^+$, let $X(w)$ be the sofic shift defined by the graph defined as follows. The graph has $|w|$ nodes $v_0, \ldots, v_{|w|-1}$. For each $i \in \{0, \ldots, |w|-1\}$, it has an edge labeled with 0 from v_i to $v_{i+1 \mod |w|}$. Additionally, for $i \in \{0, \ldots, |w|-1\}$, it has an edge labeled with 1 from v_i to $v_{i+1 \mod |w|}$ if, and only if, $w[i] = *$ (where $w = w[0]w[1]\ldots w[|w|-1]$).

Example 31 The graph of $X(w)$ for $w = 00*0**00$ is depicted in Figure 1.

The shift $X(w)$ can be described as follows: its elements are exactly all bi-infinite binary sequences of the form $\ldots v_{-2}v_{-1}v_0v_1v_2\ldots$ where, for every $i \in \mathbb{Z}$, v_i is a binary string of length $|w|$ that *matches* w in the following sense: at every position where v_i has a 1, w must have a $*$. It is equivalent to say that v_i can be obtained from w by replacing every $*$ in w either by a 0 or a 1. In the following we will call any element of $\{0,*\}^*$ a *pattern*. In the next lemma we collect several useful properties of the shifts $X(w)$.

Lemma 32

1. *For any $w \in \{0,*\}^+$, $0^{\mathbb{Z}} \in X(w)$. Hence, $X(w)$ is nonempty.*
2. *For any $w \in \{0,*\}^+$, $X(w)$ is irreducible.*
3. *For any $w \in \{0,*\}^+$, $h_{\text{top}}(X(w)) = \frac{\#_*(w)}{|w|}$ (where $\#_*(w)$ is the number of $*$'s in w).*
4. *For any $w \in \{0,*\}^+$ and any integer $k \geq 1$, $X(w^k) = X(w)$.*
5. *For any $v, w \in \{0,*\}^+$, if $|v| = |w|$ and if v can be obtained from w by replacing some $*$'s in w by 0's, then $X(v) \subseteq X(w)$.*

6. *For any $w \in \{0, *\}^+$, any integers a, b with $a \geq 2$ and $b \geq 0$, and any $v \in \Sigma^*$ with $|v| \leq |w|$, one has*
$$v \in \mathcal{L}(X(w)) \iff v \in \mathcal{L}(X(w^a 0^{|w| \cdot b}).$$

Proof:

1. Clear.
2. Clear.
3. Left to the reader (compare, e.g., Lemma 20 and Theorem 21 in Simonsen (2006)).
4. Clear.
5. Clear.
6. Since by Statements 4 and 5 of this lemma, the set $\mathcal{L}(X(w^a 0^{|w| \cdot b}))$ is a subset of $\mathcal{L}(X(w))$, the direction from right to left is clear. For the other direction, assume that $v \in \mathcal{L}(X(w))$ and $|v| \leq |w|$. Then v matches a subpattern of ww. Since $a \geq 2$, we obtain $v \in \mathcal{L}(X(w^a 0^{|w| \cdot b})$.

□

Proof of Proposition 21: The function H defined in Proposition 21 is discontinuous for example in 1^ω, even with respect to the weaker topology on $\mathbb{R}$, mentioned in Proposition 21. Note that $1^\omega = \chi_{\Sigma^*} = \chi_{\mathcal{L}(\Sigma^\mathbb{Z})}$, and that the full shift $\Sigma^\mathbb{Z} = X(*)$ has entropy 1 and is irreducible. If H were continuous in 1^ω with respect to the weaker topology on $\mathbb{R}$, then there would be a finite prefix 1^n of 1^ω with $n \geq 2$ such that the entropy of any shift X with the property that $\chi_{\mathcal{L}(X)}$ starts with 1^n would be strictly larger than $1/2$. In that case, we set $l := \lceil \log_2(n) \rceil$, and observe that by Lemma 32.3 the shift $X(*^l 0^l)$ has entropy $1/2$, but, obviously, all strings of length $\leq l$ are in the language of this shift, hence, the characteristic sequence of its language starts with 1^n as well. □

Before we come to the proof of Theorem 22, we need some more preparations. We had defined right-computable real numbers as the limits of computable, non-increasing sequences of rational numbers. In fact, one can restrict oneself to dyadic rational numbers, and one can ask that the sequence can be computed quite fast. A sequence $(z_n)_{n \in \mathbb{N}}$ of natural numbers is *computable in time polynomial in* n if there is a Turing machine which, given a binary name of some $n \in \mathbb{N}$, computes a binary name of z_n in time bounded by a polynomial in n.

Lemma 33 *A real number $\alpha \in [0, 1]$ is right-computable if, and only if, there is a sequence $(z_n)_{n \in \mathbb{N}}$ of natural numbers computable in time polynomial in n such that $z_n \in \{1, \ldots, 2^n\}$, for all n, and such that the sequence $(z_n/2^n)_{n \in \mathbb{N}}$ is non-increasing and converges to α.*

For reasons which will become clear in the proof of Theorem 22, we do not allow $z_n = 0$.

Proof: The direction from right to left is clear: for a sequence $(z_n)_{n \in \mathbb{N}}$ with the properties formulated in Lemma 33 the limit $\lim_{n \to \infty} z_n/2^n$ is a right-computable real number in $[0, 1]$.

We come to the direction from left to right. Let $\alpha \in [0,1]$ be right-computable. Let us assume that a computable, non-increasing sequence $(q_n)_{n\in\mathbb{N}}$ of rational numbers with $\lim_{n\to\infty} q_n = \alpha$ is known. The sequence $(y_n)_{n\in\mathbb{N}}$ defined by

$$y_n := \min\{i \in \{1,\dots,2^n\} \mid i/2^n \geq \min\{1, q_n\}\}$$

is a computable sequence of integers $y_n \in \{1,\dots,2^n\}$ such that the sequence $(y_n/2^n)_{n\in\mathbb{N}}$ is non-increasing and converges to α. The only problem why we cannot take y_n for z_n is that the sequence $(y_n)_{n\in\mathbb{N}}$ might not be computable in time polynomial in n. Let M be a Turing machine which, without ever stopping, writes the one-way infinite sequence

$$\mathrm{bin}(y_0)\#\mathrm{bin}(y_1)\#\mathrm{bin}(y_2)\#\dots$$

onto a one-way output tape, where, for an integer x, we denote its binary name by $\mathrm{bin}(x)$. We define the desired sequence $(z_n)_{n\in\mathbb{N}}$ as follows.

For z_n, perform the first n steps of the computation of M. Let m be the number of $\#$'s written by M during these steps. In case $m = 0$, define $z_n := 2^n$. In case $m > 0$, define $z_n := y_{m-1} \cdot 2^{n+1-m}$.

Note that $y_0 = 1$ and that for any $x, l \in \mathbb{N}$ with $x > 0$ it is clear that $\mathrm{bin}(x \cdot 2^l) = \mathrm{bin}(x)0^l$. It is clear that the sequence $(z_n)_{n\in\mathbb{N}}$ is computable in time polynomial in n. Furthermore, $z_n \in \{1,\dots,2^n\}$ for all n, and the sequence $(z_n/2^n)_{n\in\mathbb{N}}$ is non-increasing and converges to α. $\square$

The strategy of the proof of Theorem 22 is to consider a computable sequence of integers $(z_n)_{n\in\mathbb{N}}$ as in the previous lemma and to construct a decreasing sequence of (sofic) shifts X_n with entropy equal to $z_n/2^n$. Then the desired shift X will be defined as the limit of the shifts X_n, as in the following lemma.

Lemma 34 *Let $(X_n)_{n\in\mathbb{N}}$ be a sequence of nonempty subshifts of $\Sigma^{\mathbb{Z}}$ with $X_{n+1} \subseteq X_n$ for all n. Set $X := \bigcap_{n\in\mathbb{N}} X_n$. Then*

1. *X is a nonempty subshift of $\Sigma^{\mathbb{Z}}$ as well,*
2. *$\mathcal{L}(X) = \bigcap_{n\in\mathbb{N}} \mathcal{L}(X_n)$,*
3. *$h_{\mathrm{top}}(X) = \lim_{n\to\infty} h_{\mathrm{top}}(X_n)$.*

Proof: The first statement, that X is a subshift of $\Sigma^{\mathbb{Z}}$, is shown once we have shown that X is nonempty and closed and that $\sigma(X) = X$.

"X is nonempty": Consider an arbitrary sequence $(x_n)_{n\in\mathbb{N}}$ of elements $x_n \in X_n$, for each n. Since $\Sigma^{\mathbb{Z}}$ is compact, this sequence has an accumulation point in $\Sigma^{\mathbb{Z}}$. Let us call it x. Since for all $n \in \mathbb{N}$ and all $m \geq n$ we have $x_m \in X_m \subseteq X_n$, we conclude that x is contained in the closure of X_n for all n, hence in X_n itself for all n, hence $x \in X$. Thus, X is nonempty.

"X is closed": For every n, X_n is closed, and the intersection of arbitrarily many closed sets is closed again.

"$\sigma(X) \subseteq X$": From $\sigma(X_n) \subseteq X_n$ for all n we obtain $\sigma(\bigcap_{n\in\mathbb{N}} X_n) \subseteq \bigcap_{n\in\mathbb{N}} X_n$, i.e., $\sigma(X) \subseteq X$.

"$\sigma(X) \supseteq X$": Let us fix some $y \in X$. We have to show that there is some $x \in X$ with $\sigma(x) = y$. Since $y \in X_n$ and $\sigma(X_n) \supseteq X_n$, for all n, for every $n \in \mathbb{N}$ there is some $x_n \in X_n$ with $\sigma(x_n) = y$. As in the proof of "X is nonempty" we can conclude that the sequence $(x_n)_{n\in\mathbb{N}}$ has an accumulation point

x, and that x is an element of X. The sequence $(x_n)_{n\in\mathbb{N}}$ has a subsequence that converges to x. Since $\sigma(x_n) = y$ for all n and since σ is continuous, we conclude that $\sigma(x) = y$.

"$\mathcal{L}(X) \subseteq \bigcap_{n\in\mathbb{N}} \mathcal{L}(X_n)$": For any subshifts Y, Z of $\Sigma^{\mathbb{Z}}$ with $Y \subseteq Z$ one has $\mathcal{L}(Y) \subseteq \mathcal{L}(Z)$. The assertion follows.

"$\mathcal{L}(X) \supseteq \bigcap_{n\in\mathbb{N}} \mathcal{L}(X_n)$": Let us fix some $w \in \bigcap_{n\in\mathbb{N}} \mathcal{L}(X_n)$. For every $n \in \mathbb{N}$ there is some $x_n \in X_n$ such that $x_n[0\ldots|w|-1] = w$. As in the proof of "X is nonempty" we can conclude that the sequence $(x_n)_{n\in\mathbb{N}}$ has an accumulation point x, and that x is an element of X. It is also clear that $x[0,\ldots,|w|-1] = w$. This implies $w \in \mathcal{L}(X)$.

"$h_{\mathrm{top}}(X) = \lim_{n\to\infty} h_{\mathrm{top}}(X_n)$": Since for any subshifts Y, Z of $\Sigma^{\mathbb{Z}}$ with $Y \subseteq Z$ one has $h_{\mathrm{top}}(Y) \le h_{\mathrm{top}}(Z)$, the sequence $h_{\mathrm{top}}(X_n)$ is non-increasing. Since it is bounded from below by 0, the limit $\lim_{n\to\infty} h_{\mathrm{top}}(X_n)$ exists. Furthermore, $h_{\mathrm{top}}(X) \le h_{\mathrm{top}}(X_n)$, for all n, follows as well, hence $h_{\mathrm{top}}(X) \le \lim_{n\to\infty} h_{\mathrm{top}}(X_n)$. We still have to show that $h_{\mathrm{top}}(X)$ cannot be strictly smaller than $\lim_{n\to\infty} h_{\mathrm{top}}(X_n)$. In order to show this, let us fix some $\varepsilon > 0$. We show that $h_{\mathrm{top}}(X_n) \le h_{\mathrm{top}}(X) + \varepsilon$ for almost all n. Remember that for an arbitrary subshift Y of $\Sigma^{\mathbb{Z}}$,

$$h_{\mathrm{top}}(Y) = \lim_{k\to\infty} \frac{\log_2 |\mathcal{L}(Y) \cap \Sigma^k|}{k} = \inf_{k\ge 1} \frac{\log_2 |\mathcal{L}(Y) \cap \Sigma^k|}{k}.$$

Therefore, there exists some $k \ge 1$ with $\frac{\log_2 |\mathcal{L}(X)\cap\Sigma^k|}{k} \le h_{\mathrm{top}}(X) + \varepsilon$. Above we have seen $\mathcal{L}(X) = \bigcap_{n\in\mathbb{N}} \mathcal{L}(X_n)$. Especially, $\mathcal{L}(X) \cap \Sigma^k = \bigcap_{n\in\mathbb{N}} (\mathcal{L}(X_n) \cap \Sigma^k)$. Since the sets $\mathcal{L}(X_n) \cap \Sigma^k$ are finite and $\mathcal{L}(X_m) \cap \Sigma^k \supseteq \mathcal{L}(X_n) \cap \Sigma^k$ for any $m, n \in \mathbb{N}$ with $m \le n$, we conclude that there is some $l \in \mathbb{N}$ such that for all $n \ge l$ we have $\mathcal{L}(X_n) \cap \Sigma^k = \mathcal{L}(X) \cap \Sigma^k$. We conclude that for all $n \ge l$

$$h_{\mathrm{top}}(X_n) \le \frac{\log_2 |\mathcal{L}(X_n) \cap \Sigma^k|}{k} = \frac{\log_2 |\mathcal{L}(X) \cap \Sigma^k|}{k} \le h_{\mathrm{top}}(X) + \varepsilon.$$

This ends the proof. □

Proof of Theorem 22: Let $\alpha \in [0,1]$ be a right-computable real number. By Lemma 33 there exists a sequence $(z_n)_{n\in\mathbb{N}}$ of integers computable in time polynomial in n with $z_n \in \{1,\ldots,2^n\}$ such that the sequence $(q_n)_{n\in\mathbb{N}}$ defined by $q_n := z_n/2^n$ is non-increasing and converges to α. We define a sequence $(w_n)_{n\in\mathbb{N}}$ of patterns $w_n \in \{0,*\}^+$ recursively by

$$\begin{aligned} w_0 &:= *, \\ w_{n+1} &:= w_n^{2z_{n+1}} 0^{|w_n|\cdot(4z_n - 2z_{n+1})}. \end{aligned}$$

Note that $z_n \neq 0$ for all n and that the assumption that the sequence $(q_n)_{n\in\mathbb{N}}$ is non-increasing implies that $4z_n \ge 2z_{n+1}$ for all n. We define a sequence $(X_n)_{n\in\mathbb{N}}$ of sofic shifts by

$$X_n := X(w_n).$$

By Lemma 32.1, X_n is nonempty, for each n. Since w_{n+1} can be obtained from $w_n^{4z_n}$ by replacing some $*$'s in $w_n^{4z_n}$ by 0's, Lemma 32.4 and 32.5 tell us that $X(w_{n+1}) \subseteq X(w_n)$, for all n. We define

$$X := \bigcap_{n\in\mathbb{N}} X_n.$$

By Lemma 34.1, X is a nonempty subshift of $\Sigma^{\mathbb{Z}}$. We claim that it has all desired properties.

"$h_{\mathrm{top}}(X) = \alpha$": This is true by Lemma 34.3 because we have constructed X_n so that $h_{\mathrm{top}}(X_n) = q_n$ for all n. We show this by induction. For $n = 0$ it is true because $z_0 = 1$, hence $q_0 = 1$, and $X_0 = X(w_0) = X(*) = \Sigma^{\mathbb{Z}}$ is the full shift, and the entropy of the full shift is equal to 1. For $n + 1$ we obtain by Lemma 32.3 and by the induction hypothesis

$$\begin{aligned} h_{\mathrm{top}}(X_{n+1}) &= \frac{\#_*(w_{n+1})}{|w_{n+1}|} \\ &= \frac{\#_*(w_n) \cdot 2z_{n+1}}{|w_n| \cdot 4z_n} \\ &= h_{\mathrm{top}}(X_n) \cdot \frac{2z_{n+1}}{4z_n} \\ &= q_n \cdot \frac{z_{n+1}}{2z_n} \\ &= \frac{z_n}{2^n} \cdot \frac{z_{n+1}}{2z_n} \\ &= \frac{z_{n+1}}{2^{n+1}} \\ &= q_{n+1}. \end{aligned}$$

Thus, the subshift X has the desired entropy.

We still have to show that X is irreducible and that its language $\mathcal{L}(X)$ is decidable in polynomial time.

First, we observe that $|w_0| = 1$ and $|w_{n+1}| = |w_n| \cdot 4 \cdot z_n \geq |w_n| \cdot 4$ for all n, hence $|w_n| \geq 4^n$ for all n.

Secondly, we claim that for any $n \in \mathbb{N}$ and any $v \in \Sigma^*$ with $|v| \leq |w_n|$

$$v \in \mathcal{L}(X(w_n)) \iff v \in \mathcal{L}(X). \tag{1}$$

Indeed, by Lemma 34.2 $\mathcal{L}(X) = \bigcap_{m \in \mathbb{N}} \mathcal{L}(X(w_m))$. Thus, the direction from right to left in (1) is clear. For the direction from left to right, fix some $n \in \mathbb{N}$ and some $v \in \mathcal{L}(X(w_n))$ with $|v| \leq |w_n|$. By Lemma 32.6, $v \in \mathcal{L}(X(w_{n+1}))$. Since $|v| \leq |w_n| \leq |w_{n+1}|$, we can repeat this argument and obtain $v \in \mathcal{L}(X(w_{n+2}))$, and so on. Thus, $v \in \mathcal{L}(X(w_m))$ for all $m \geq n$. Since $v \in \mathcal{L}(X(w_m))$ for all $m < n$ is clear anyway (due to $\mathcal{L}(X(w_k)) \supseteq \mathcal{L}(X(w_{k+1}))$ for all $k \in \mathbb{N}$), we obtain $v \in \mathcal{L}(X)$. This ends the proof of (1).

"X is irreducible": Assume that $x, z \in \Sigma^*$ are two strings in $\mathcal{L}(X)$. Choose n large enough so that $|w_n| \geq \max\{|x|, |z|\}$. By Lemma 34.2, we observe $x, z \in \mathcal{L}(X(w_n))$. This implies that x matches a subpattern of $w_n w_n$ in the sense explained above, i.e., in the sense that it can be obtained from a subpattern of $w_n w_n$ if the $*$'s in the subpattern are replaced appropriately by 0's or 1's. The same as for x is true for z. But then, since w_{n+1} contains $w_n w_n$ as a subpattern, there is a string $y \in \Sigma^*$ such that xyz matches a subpattern of $w_{n+1} w_{n+1}$ in the same sense, and, hence, by the same argument, matches a subpattern of w_{n+2}. By (1), this implies that xyz is an element of $\mathcal{L}(X)$. Hence, X is irreducible.

"$\mathcal{L}(X)$ is decidable in polynomial time": We claim that the following algorithm decides for an arbitrary input string $v \in \Sigma^*$ whether v belongs to $\mathcal{L}(X)$ or not. Furthermore, the algorithm can be implemented on a Turing machine that works in time polynomial in $|v|$. In the algorithm we use a Turing machine M which, given a binary name of any $n \in \mathbb{N}$, computes in time polynomial in n the binary name of z_n.

Algorithm: Assume that some $v \in \Sigma^*$ is the input.
First part: For $i = 0, 1, 2, \dots$ compute the binary name of z_i (use M for this task) and the string w_i until a number i has been found such that $|w_i| \geq |v|$.
Second part: For this i, check whether v is an element of $\mathcal{L}(X(w_i))$ by checking whether v matches a subpattern of $w_i w_i$.
(*End of the description of the algorithm*)

Due to $|w_n| \geq 4^n$, the search in the first part of the algorithm will be successful, and by (1) it is clear that this algorithm decides whether v is an element of $\mathcal{L}(X)$ or not. We show now that the algorithm (when implemented properly on a Turing machine) works in time polynomial in $|v|$.

Let i be the smallest number i with $|w_i| \geq |v|$. Since $|w_n| \geq 4^n$ for all n, we obtain $i = 0$ for $|v| \leq 1$ and $i \leq \lceil \log_4(|v|) \rceil \leq 1 + \log_4(|v|)$ for $|v| > 1$. The computation of the numbers $z_0, \dots, z_i$ in the first part of the algorithm can be done in time polynomial in i, hence, certainly in time polynomial in $|v|$. If, for some $k < i$, the string w_k and the numbers z_k, z_{k+1} are known, one can in time polynomial in the length of w_{k+1} compute the string w_{k+1}. Note that $w_0 = *$. Hence, if $|v| \leq 1$, the algorithm will already stop with $i = 0$. Assume now that $|v| > 1$. Then $|w_0| = |*| = 1 < |v|$, hence $i \geq 1$ and $|w_{i-1}| < |v|$, hence,

$$|w_i| = |w_{i-1}| \cdot 4 \cdot z_{i-1} \leq |w_{i-1}| \cdot 4 \cdot 2^{i-1} < |v| \cdot 4 \cdot 2^{\log_4(|v|)} = 4 \cdot |v| \cdot \sqrt{|v|}.$$

This and $|w_k| \leq |w_i|$ for $k \leq i$ imply that the strings $w_0, \dots, w_i$ can be computed in time polynomial in $|v|$ (in the first part of the algorithm). Thus, the first part of the algorithm works in time polynomial in $|v|$. The checking in the second part can be done in detail as follow: for each position in $w_i w_i$, one checks whether v matches the subpattern of $w_i w_i$ starting at this position with length $|v|$ (of course, at the last $|v| - 1$ positions of $w_i w_i$ no subpattern of length $|v|$ can start). For each position, this checking can be done in time linear in $|v|$. Since there are only $2 \cdot |w_i|$ positions, the second part takes only time polynomial in $|v|$. Thus, the whole algorithm works in time polynomial in $|v|$. This ends the proof of Theorem 22. □

Proof of Proposition 16.2, 2nd variant: For any real number $\alpha \in [0, 1]$ there exists a sequence $(z_n)_{n \in \mathbb{N}}$ of numbers z_n with $z_n \in \{1, \dots, 2^n\}$, for each n, such that the sequence $(z_n / 2^n)_{n \in \mathbb{N}}$ is non-increasing and converges to α. By defining patterns w_n and shifts $X(w_n)$ and, finally, a shift X in exactly the same way as in the proof of Theorem 22, one obtains a nonempty, irreducible shift X with topological entropy α. □

5 Summary and Open Problems

We have answered positively the question by Simonsen (2006) whether there exists a shift with decidable language whose topological entropy is non-computable. In fact, the shift constructed by us is even irreducible and its language is polynomial time decidable. Simonsen (2006) also suggested to study the class of shifts with primitive recursive language, a subclass of the shifts with decidable language. Since any polynomial time language is primitive recursive, our result shows that restriction to shifts with primitive recursive language does not help with respect to the topological entropy: there are even shifts in this smaller class whose topological entropy is non-computable.

In the second paragraph of the introduction we mentioned several positive results concerning the computability of the entropy for certain types of shift spaces. But there is still a gap between those positive results and the negative result of this paper. It remains an interesting task to narrow this gap and to identify more known classes of shifts such that the entropy is computable for the shifts in the class.

On the other hand, we have also shown that computability of the entropy of a shift does not imply any kind of computability of the language of the shift. Especially, it does not imply that the language is decidable, not even if the language happens to be computably co-enumerable. Thus, in a computability theoretic sense the language of a shift and the entropy of the shift are not closely related: decidability of the language does not imply computability of the entropy, and computability of the entropy does not imply decidability of the language.

We have not only shown that there exist shifts with (polynomial time) decidable language and non-computable entropy. Actually, we have classified exactly the real numbers that can appear as the topological entropy of a subshift of the full shift of all binary bi-infinite sequences with co-c.e. resp. with decidable resp. with polynomial decidable language (the answer is in each case: exactly the right-computable real numbers between 0 and 1). Note that Hochman and Meyerovitch (2007) showed an analogous result for multi-dimensional shifts of finite type: they showed that the real numbers that are the entropy of a multi-dimensional shift of finite type are exactly the non-negative right-computable real numbers. It would be interesting to ask the same question for other types of dynamical systems, modifying slightly two questions raised by Koiran (2001): which real numbers can be the entropy of a one-dimensional cellular automaton (note that Hurd et al. (1992) have shown that there is no algorithm which, given a one-dimensional cellular automaton, computes its entropy with any desired precision) or of a piecewise affine map as studied by Koiran (2001)?

Acknowledgements

We thank one of the anonymous referee for informing us about the paper by Simonsen (2005) and the preprint by Hochman and Meyerovitch (2007).

References

P. Hertling and C. Spandl. Computability theoretic properties of the entropy of gap shifts. *Fundamenta Informaticae*, 83(1–2):141–157, 2008.

M. Hochman and T. Meyerovitch. A characterization of the entropies of multidimensional shifts of finite type. arXiv:math/0703206v1, March 2007.

L. Hurd, J. Kari, and K. Culik. The topological entropy of cellular automata is uncomputable. *Ergodic Theory and Dynamical Systems*, 12:255–265, 1992.

P. Koiran. The topological entropy of iterated piecewise affine maps is uncomputable. *Discrete Mathematics and Theoretical Computer Science*, 4(2):351–356, 2001.

D. Lind and B. Marcus. *Symbolic Dynamics and Coding*. Cambridge University Press, Cambridge, UK, 1995.

J. Milnor. Is entropy effectively computable? Remark, see http://www.math.sunysb.edu/∼jack/comp-ent.pdf, 2002.

J. G. Simonsen. On beta-shifts having arithmetical languages. In J. Jedrzejowicz and A. Szepietowski, editors, *Mathematical Foundations of Computer Science 2005, 30th International Symposium, MFCS 2005, Gdansk, Poland, August 29 - September 2, 2005, Proceedings*, volume 3618 of *Lecture Notes in Computer Science*, pages 757–768. Springer, 2005.

J. G. Simonsen. On the computability of the topological entropy of subshifts. *Discrete Mathematics and Theoretical Computer Science*, 6:83–96, 2006.

C. Spandl. Computing the topological entropy of shifts. *Mathematical Logic Quarterly*, 53(4–5):493–510, 2007.

K. Weihrauch. *Computable Analysis*. Springer, Berlin, 2000.

X. Zheng. Recursive approximability of real numbers. *Mathematical Logic Quarterly*, 48(Suppl. 1): 131–156, 2002.