On the Complexity of Computing the Capacity of Codes That Avoid Forbidden Difference Patterns

Vincent D. Blondel, Raphaël Jungers, and Vladimir Protasov

Abstract—Some questions related to the computation of the capacity of codes that avoid forbidden difference patterns are analysed. The maximal number of n-bit sequences whose pairwise differences do not contain some given forbidden difference patterns is known to increase exponentially with n; the coefficient of the exponent is the capacity of the forbidden patterns. In this paper, new inequalities for the capacity are given that allow for the approximation of the capacity with arbitrary high accuracy. The computational cost of the algorithm derived from these inequalities is fixed once the desired accuracy is given. Subsequently, a polynomial time algorithm is given for determining if the capacity of a set is positive while the same problem is shown to be NP-hard when the sets of forbidden patterns are defined over an extended set of symbols. Finally, the existence of extremal norms is proved for any set of matrices arising in the capacity computation. Based on this result, a second capacity approximating algorithm is proposed. The usefulness of this algorithm is illustrated by computing exactly the capacity of particular codes that were only known approximately.

Index Terms—Approximation algorithm, capacity of codes, coding theory, complexity of capacity, joint spectral radius, NP-hardness.

I. INTRODUCTION

In certain coding applications, one is interested in binary codes whose elements avoid a set of forbidden patterns. In order to minimize the error probability of some particular magnetic-recording systems, a more complicated problem arises when it is desirable to find code words whose *differences* avoid forbidden patterns.

Let $\{0,1\}^n$ denote the set of words of length n over $\{0,1\}$ and let $u, v \in \{0,1\}^n$. The difference $u - v$ is a word of length n over $\{-1, 0, +1\}$ (as a shorthand, we will use $\{-, 0, +\}$ instead of $\{-1, 0, +1\}$). The difference $u - v$ is obtained from u and v by symbol-by-symbol subtraction so that, for example, $0110 - 1011 = -+0-$. Consider now a finite set D of words over $\{-, 0, +\}$; we think of D as a set of forbidden difference patterns. A set (or *code*) $C \subseteq \{0,1\}^n$ is said to *avoid* the set D if none of the differences of words in C contain a word from D as subword, that is, none of the differences $u - v$ with $u, v \in C$ can be written as $u - v = xdy$ for $d \in D$ and some (possibly empty) words x and y over $\{-, 0, +\}$.

We are interested in the largest cardinality, which we denote by $\delta_n(D)$, of sets of words of length n whose differences avoid the forbidden patterns in D. If the set D is empty, then there are no forbidden patterns and $\delta_n(D) = 2^n$. When D is nonempty, then $\delta_n(D)$ grows exponentially with the word length n and is asymptotically equal to $2^{\mathrm{cap}(D)n}$ where the scalar $0 \leq \mathrm{cap}(D) \leq 1$ is the *capacity* of the set D. The capacity is thus a measure of how constraining a set D

is: the smaller the capacity, the more constraining are the forbidden difference patterns.

As an illustration consider the set of forbidden patterns $D = \{+-, ++\}$. Differences between two words in $C = \{u_1 0 u_2 0 \ldots 0 u_k : u_i \in \{0,1\}\}$ will have a "0" in any succession of two characters and will therefore not contain any of the forbidden patterns. From this it follows that $\delta_n \geq 2^{\lceil n/2 \rceil}$ and so $\mathrm{cap}(D) \geq 1/2$. One can show in fact that $\mathrm{cap}(D) = 1/2$. This follows from the next proposition combined with the simple observation that the capacity of the set $D = \{+-, ++\}$ is identical to the capacity of the set $D = \{+-, ++, -+, --\}$.

Proposition 1: The capacity of the set $\{+, -\}^m$ is given by $(m-1)/m$.

Proof: Let C_{km} be a code of length km avoiding D. In any given window of length m, the set of words appearing cannot contain both u and $\bar{u}$ (we use $\bar{u}$ to denote the word obtained by inverting the ones and the zeros in u). This implies that there are at most 2^{m-1} different words in any given window of size m. Let us now consider words in C_{km} as a concatenation of k words of length m. There are at most $2^{(m-1)k}$ words in C_{km} and so $\mathrm{cap}(D) \leq (m-1)/m$. Now consider the code

$$C_{km} = \{z_1 0 z_2 0 \ldots 0 z_k : z_i \in \{0,1\}^{m-1}\}. \tag{1}$$

This code satisfies the constraints, and so the bound $(m-1)/m$ is reached.□

The computation of the capacity is not always that easy. As an example it is proved in [15] that the capacity of $\{+++\}$ is given by $\log_2((1 + (19 + 3\sqrt{33})^{1/3} + (19 - 3\sqrt{33})^{1/3})/3) = 0.8791\ldots$ and the same reference provides numerical bounds for the capacity of $\{0+-+\}$ for which no explicit expression is known.

The capacity of codes that avoid forbidden difference patterns was first introduced and studied by Moision, Orlitsky, and Siegel. In [15], these authors provide explicit values for the capacity of particular sets of forbidden patterns and they prove that, in general, the capacity of a forbidden set D can be obtained as the logarithm of the joint spectral radius of a set of matrices that have binary entries. The joint spectral radius, which we formally define below, is a quantity that quantifies the maximal asymptotic growth rate of products of matrices taken from a set. This quantity is notoriously difficult to compute and to approximate. It is known in particular that the problem of computing, or even approximating, the joint spectral radius of two matrices with binary entries is a problem that is NP-hard [20] and that the problem of determining if the joint spectral radius of two matrices with nonnegative entries is greater than one is undecidable [6]. In addition to this, the size of the matrices constructed in [15] for computing the capacity is not polynomial in the size of the forbidden set D and so even the construction of the set of matrices is an operation that cannot be performed in polynomial time. However, as pointed out in [15], the matrices that arise in the context of capacity computation have a particular structure and so the capacity could very well be computable in polynomial time.

We provide several results in this paper. All are related to the capacity computation and its complexity.

We first provide new bounds that relate the capacity of a set of forbidden patterns D with the values $\delta_n(D)$, the maximum size of a code of length n avoiding D. These bounds depend on parameters that express the number and positions of zeros in the patterns of D. These new bounds allow us to compute the capacity of any set to any given degree of accuracy by numerically evaluating $\delta_n(D)$ for some value of n. The approximation algorithm resulting from these bounds has exponential growth but provides an *a priori* guaranteed precision, and so the computational effort required to compute the capacity to a given

degree of accuracy can be evaluated before the calculations are actually performed. As an example, it follows from the bounds we provide that the capacity of a set of forbidden patterns that does not contain any 0's can be computed with an accuracy of 90% by evaluating $\delta_n(D)$ for $n = 10$ (see Corollary 2).

In Sections II–VII, we provide explicit necessary and sufficient conditions for a set to have positive capacity and we use this condition for producing a polynomial time algorithm that decides whether or not the capacity of a set is positive. As explained above, the capacity of a set is given by the logarithm of the joint spectral radius of a set of matrices constructed from the forbidden patterns. Our polynomial time algorithm therefore provides a procedure for checking whether or not the joint spectral radius of these matrices is larger than one. This problem is known to be undecidable for general matrices.

We then consider the situation where in addition to the forbidden symbols $-$, 0, and $+$ the forbidden patterns in D may also include the symbol $\pm$, where $\pm$ stands for both the symbol $+$ and $-$. We prove that in this case the problem of computing the capacity, or even determining if this capacity is positive, becomes NP-hard.

Finally, we show an algebraic property of the sets of matrices constructed in order to compute the capacity: there always exists an extremal norm for this set. This theoretical result makes it possible to apply specific algorithms in order to compute the joint spectral radius. These methods, although non-polynomial, can be more efficient than general ones, and we use them for computing the capacity of some specific forbidden sets.

These results allow us to better delineate the capacity computation problems that are polynomial time solvable from those that are not. We do however not provide in this paper an answer to the question, which was the original motivation for the research reported here, as to whether or not one can compute the capacity of sets of forbidden patterns over $\{-, 0, +\}$ in polynomial time. This interesting question that was already raised in [15], remains unsettled.

II. CAPACITY AND JOINT SPECTRAL RADIUS

Let D be a set of forbidden patterns over the alphabet $\{-, 0, +\}$ and consider for any $n \geq 1$ the largest cardinality, denoted by $\delta_n(D)$, of sets of words of length n whose pairwise differences avoid the forbidden patterns in D. The capacity of D is defined by

$$\mathrm{cap}(D) = \lim_{n \to \infty} \frac{\log_2 \delta_n(D)}{n}. \tag{2}$$

Moision *et al.* show in [15] how to represent codes as products of matrices. Associated to any set D of forbidden patterns, they construct a finite set $\Sigma(D)$ of matrices for which

$$\delta_{m-1+n} = \max\{\|A_1 \ldots A_n\| : A_i \in \Sigma(D)\}. \tag{3}$$

In this expression, the matrix norm used is the sum of the absolute values of the matrix entries. For sake of conciseness, we do not reproduce here the explicit construction of the matrices of $\Sigma(D)$ but refer instead the interested reader to [15] for more details. Let us comment here on the number and size of the matrices in $\Sigma(D)$; these issues are relevant for the questions raised in this correspondence. If the forbidden patterns in D have identical length m, then the number of matrices in $\Sigma(D)$ can be doubly exponential in m and all matrices in $\Sigma(D)$ have dimension $2^{m-1} \times 2^{m-1}$. If the forbidden patterns in D have different lengths, then one can construct a set D' whose forbidden patterns have equal length and for which $\mathrm{cap}(D) = \mathrm{cap}(D')$. Unfortunately, the number of patterns in D' can grow exponentially with the size of D so that the number of matrices in the set $\Sigma(D)$ constructed in [15] is in

fact even worse than in the former case. Capacity approximation algorithms based on the direct computation of the set $\Sigma(D)$ will therefore not be tractable even for small sets D.

We now describe the connection between the capacity of D and the joint spectral radius of $\Sigma(D)$. Combining (3) and (2) we deduce that

$$
\begin{aligned}
\mathrm{cap}(D) &= \lim_{n \to \infty} \frac{\log_2 \delta_n(D)}{n} \\
&= \lim_{n \to \infty} \frac{\log_2 \delta_{m-1+n}(D)}{m-1+n} \\
&= \lim_{n \to \infty} \frac{\log_2 \max_{A_i \in \Sigma} \|A_1 \ldots A_n\|}{n} \\
&= \log_2 \lim_{n \to \infty} \max_{A_i \in \Sigma} \|A_1 \ldots A_n\|^{1/n}.
\end{aligned}
$$

The quantity $\lim_{n \to \infty} \max_{A_i \in \Sigma} \|A_1 \ldots A_n\|^{1/n}$ appearing in the last identity is a joint spectral radius. For any compact set of matrices A, the joint spectral radius of A is defined by

$$\rho(A) = \limsup_{n \to \infty} \sup_{A_i \in A} \|A_1 \ldots A_n\|^{1/n}.$$

Hence, we have the fundamental relation

$$\mathrm{cap}(D) = \log_2 \rho(\Sigma(D)).$$

The joint spectral radius of a set of matrices is a quantity that was introduced by Rota and Strang [19] and that has received intense research attention in the last decade. For more references on the joint spectral radius, consult the survey [5].

III. UPPER AND LOWER BOUNDS

In this section, we derive bounds that relate the capacity of a set D with $\delta_n(D)$. Consider some set D of forbidden patterns and denote by r_1 (respectively, r_2) the maximal k for which 0^k is the prefix (respectively, suffix) of some pattern in D. No pattern in D begins with more than r_1 zeros and no pattern in D ends with more than r_2 zeros. We also denote by r the maximal number of consecutive zeros in any pattern in D; obviously, $r \geq \max(r_1, r_2)$. In the next theorem we provide upper and lower bounds on the capacity $\mathrm{cap}(D)$ in terms of $\delta_n(D)$.

Theorem 1: For any $n \geq r_1 + r_2$ we have

$$\frac{\log_2 \delta_n(D) - (r_1 + r_2)}{n + r + 1 - (r_1 + r_2)} \leq \mathrm{cap}(D) \leq \frac{\log_2 \delta_n(D)}{n}. \tag{4}$$

Proof: Let us first consider the upper bound. The following equation is straightforward, given any positive integers k, n, and any set of forbidden patterns D

$$\delta_{kn} \leq \delta_n^k. \tag{5}$$

Indeed, considering any word of length kn as the concatenation of k subwords of length n, for each of these subwords we have at most δ_n possibilities. Taking the $\frac{1}{kn}$th power of both sides of this inequality and taking the limit $k \to \infty$, we obtain

$$2^{\mathrm{cap}(D)} \leq \delta_n^{1/n}$$

$$\rho^n \leq \delta_n.$$

Now let us consider the lower bound. The optimal code of length n contains at least $\left\lceil 2^{-r_1-r_2} \delta_n(D) \right\rceil$ words that coincide in the first r_1 bits and in the last r_2 bits (because there are in total $2^{r_1+r_2}$ different words of length $r_1 + r_2$). Denote the set of strings of all these words from $(r_1 + 1)$st bit to $(n - r_2)$th bit by C'. This set contains at least $\left\lceil 2^{-r_1-r_2} \delta_n(D) \right\rceil$ different words of length $n - r_1 - r_2$. Then for any $l \geq 1$ the code

$$C = \left\{ u_1 0^{r+1} u_2 0^{r+1} \ldots 0^{r+1} u_l 0^{r+1}, \quad u_k \in C', k = 1, \ldots, l \right\}$$

avoids D. The cardinality of this code is at least $\left\lceil 2^{-r_1-r_2}\delta_n(D)\right\rceil^l$ and the length of its words is $N = l(n - r_1 - r_2 + r + 1)$. Therefore, for any l we have

$$\delta_N(D) \geq \left\lceil 2^{-r_1-r_2}\delta_n(D)\right\rceil^l.$$

Taking the power $1/N$ of both sides of this inequality, we get

$$\left[\delta_N(D)\right]^{1/N} \geq \left\lceil 2^{-r_1-r_2}\delta_n(D)\right\rceil^{1/(n-r_1-r_2+r+1)}$$

which as $N \to \infty$ yields

$$\rho \geq \left\lceil 2^{-r_1-r_2}\delta_n(D)\right\rceil^{1/(n-r_1-r_2+r+1)}.$$

Now after elementary simplifications we arrive at the lower bound on $\mathrm{cap}(D)$. □

Both bounds in Theorem 1 are sharp in the sense that they are both attained for particular sets D. The upper bound is attained for the set $D = \emptyset$ and the lower bound is attained, for instance, for the set $D = \{0^{m-1}+\}$. Indeed, in this case $r = r_1 = m - 1$, $r_2 = 0$, and $\mathrm{cap}(D) = 0$, while $\delta_n = 2^{m-1}$ for $n \geq m - 1$. Here is a direct proof of this equality, drawn from [15]. Clearly, for all $n > m - 1$, we can construct a code of size $\delta_n = 2^{m-1}$. It happens that for any given length n this size is maximum. Otherwise, there must be two *different* words u and v whose prefixes of length k coincide. In order to avoid the forbidden pattern, the $k + 1$th symbols must also be equal, and so on. But then both words are equal, and we have reached a contradiction.

Corollary 1: Let D be given and let r, r_1 and r_2 be defined as above. Then

$$\frac{\log_2 \delta_n(D)}{n} - \frac{1}{n}\max(r_1 + r_2, r + 1) \leq \mathrm{cap}(D) \leq \frac{\log_2 \delta_n(D)}{n}.$$

Proof: Simply use the fact that the capacity is always less than one. □

These bounds can be used to design an approximation algorithm that computes the capacity to any desired accuracy by evaluating δ_n for sufficiently large values of n. In contrast to previously known algorithms this algorithm has guaranteed computational cost: once the desired accuracy is given, the corresponding computational cost can easily be computed. As an illustration, consider the case of a set D for which $r_1 = r_2 = 2$ and $r = 4$. Then, by Corollary 1

$$\frac{\log_2 \delta_n(D)}{n} - \frac{5}{n} \leq \mathrm{cap}(D) \leq \frac{\log_2 \delta_n(D)}{n} \qquad (6)$$

and we can use $\log_2 \delta_n(D)/n$ as an estimate for $\mathrm{cap}(D)$ and choose a value of n for which (6) provides satisfactory accuracy bounds.

The easiest way of computing δ_n is to apply (3), by evaluating the maximum-normed product of length $n - m + 1$ of matrices taken in the set Σ. Moision *et al.* mention in [13] an improvement of this brute force method: The main idea is to compute successively some sets of matrices $\bar{\Sigma}_l$, $l = 1, 2\ldots$, with $\bar{\Sigma}_1 = \Sigma$. These are sets of products of length l, obtained by computing iteratively all products of a matrix in $\bar{\Sigma}_{l-1}$ with a matrix in Σ, and then removing from the set $\bar{\Sigma}_l$ a matrix A, if it is dominated by another matrix B in this set, that is, if each entry of A is less or equal than the corresponding entry of B. For more information about this algorithm, we refer the reader to [13]. We propose here an improvement of this method: given the set $\bar{\Sigma}_l$, one can directly compute a set $\bar{\Sigma}_{2l}$ by computing the set $\bar{\Sigma}_l^2$ and then removing from this set all matrices that are dominated. This small modification of the algorithm has dramatically improved the computational time for all the examples on which we have used it.

We may specialize the general bounds of Theorem 1 to sets of particular interest.

Corollary 2: Let D be given and let r, r_1 and r_2 be defined as above. Then

1) If $\mathrm{cap}(D) = 0$ the size of any code avoiding D is bounded above by the constant $2^{r_1+r_2}$.
2) If the patterns in D contain no zeros, then

$$n\mathrm{cap}(D) \leq \log_2 \delta_n(D) \leq (n + 1)\mathrm{cap}(D).$$

3) If none of the patterns in D start or end with a zero, then
$$n\mathrm{cap}(D) \leq \log_2 \delta_n(D) \leq (n + r + 1)\mathrm{cap}(D).$$

IV. POSITIVE CAPACITY CAN BE DECIDED IN POLYNOMIAL TIME

As previously seen by a direct argument, the capacity of the set $\{0^{m-1}+\}$ is equal to 0. In this section, we provide a systematic way of deciding when the capacity of a set is equal to zero. We first provide a simple positivity criterion that can be verified in finite time and then exploit this criterion for producing a positivity checking algorithm that runs in polynomial time. In the sequel, we will use the notation $-D$ to denote the set of elements that are the opposites to the elements of D, for example if $D = \{-+0, 0--\}$ then $-D = \{+-0, 0++\}$.

Theorem 2: Let D be a set of forbidden patterns of lengths at most m. Then $\mathrm{cap}(D) > 0$ if and only if there exists a word on the alphabet $\{+, -, 0\}$ that does not contain any word of $D \cup -D$ as subword and that has a prefix 0^m and a suffix $+0^{m-1}$.

Proof: If the capacity is positive, then for sufficiently large n there is a code avoiding D of size $\geq 2^{2m-1} + 1$. This code has at least two words u, v with the same m-bit prefix and $(m - 1)$-bit suffix (because there are in total 2^{2m-1} different words of the length $2m - 1$). Taking the difference $u - v$ and removing, if necessary, several last zeros we get an admissible string with a prefix 0^m and a suffix $\pm 0^{m-1}$. If the first bit in the suffix is $+$, then the proof is completed, if it is $-$, then we apply the same reasoning to $v - u$.

Conversely, suppose there exists a feasible string d of $+$, $-$ and 0 of some length n. Clearly, $n \geq 2m$. Let $u_1, u_0 \in \{0,1\}^{n-2m+1}$ be binary words, for which $0^m(u_1 - u_0)0^{m-1} = d$. Then for any $l \geq 1$ the code

$$C = \left\{u_{i_1}0^{m-1}\ldots u_{i_l}0^{m-1}, i_k \in \{0,1\}, k = 1, \ldots, l\right\} \qquad (7)$$

avoids D. The cardinality of this code is 2^l. Hence $\delta_n(D)$ is unbounded and we conclude from Corollary 1 that the capacity is positive. □

Corollary 3: If every word in D contains at least two nonzero symbols, then $\mathrm{cap}(D) > 0$.

Proof: For any such set the word $d = 0^m + 0^{m-1}$ is admissible, and by Theorem 2 the capacity is positive. □

Corollary 4: If D consists of one forbidden pattern p of length m, then its capacity is zero if and only if p has at least $m - 1$ consecutive 0's.

Proof: If a pattern p is 0^m or $+0^{m-1}$, then obviously there are no admissible strings, and by Theorem 2 the capacity is zero. The same holds for -0^{m-1}, since this is the negation of $+0^{m-1}$ and for $0^{m-1}\pm$ because of the symmetry. In all the other cases the admissible string exists and so $\mathrm{cap}(D) > 0$. Indeed, if p has a unique nonzero bit, then the word $d = 0^m + +0^{m-1}$ is admissible, if it has at least two nonzero bits, then the proof follows from Corollary 3. □

We now prove the polynomial-time solvability of the problem of determining whether the capacity of a set D is positive. The proof is constructive and is based on the so-called *Aho–Corasick* automaton that checks whether a given text contains as a subsequence a pattern taken from a given set [1]. Let P be a given set of patterns. The transition graph of the Aho–Corasick automaton for the set P is defined as follows. First, construct the *retrieval tree*, or trie, of the set P. The trie of P is the directed tree of which each vertex has a label representing a prefix of a pattern in P, and all prefixes are represented, including the patterns themselves. The label of the root of the tree is the empty string. Edges have a label too, which is a symbol of the used alphabet. There

is an edge labeled with the symbol a from a vertex s to a vertex t if t is the concatenation sa.

In order to have an automaton, we complete the trie by adding edges so that for each vertex s, and each symbol a, there is an edge labeled a leaving s. This edge points to the vertex of the trie of which the label is the longest suffix of the concatenation sa. Note that this vertex can be the root (that is, the empty string) if no vertex in the trie is a suffix of sa. Finally, the accepting states of the automaton are the vertices whose labels are patterns of P. This automaton accepts words that contain a pattern in P and halts whenever this pattern is a suffix of the entered text.

If $0^k \in D$ or $+0^k \in D$, $k \leq m$, then, by Theorem 2, $\mathrm{cap}(D) = 0$. If this is not the case, we construct the graph of the automaton of Aho–Corasick for the set $P = D \cup (-D) \cup \{+0^{m-1}\}$. We then remove any vertex labeled with a pattern in P (i.e., a state reached when a suffix of the text entered is in the set P) except the vertex labeled $\{+0^{m-1}\}$. The size of the constructed graph is polynomial in the size and the number of the forbidden patterns. Let us now denote q_{0^m} the state reached after entering the word 0^m. This state is well defined since 0^m does not contain any forbidden pattern, and hence no state reached after entering any prefix of the string 0^m was removed from the primary automaton. We also denote $q_{+0^{m-1}}$ the state corresponding to the suffix $+0^{m-1}$ for the entered text (i.e., the accepting state corresponding to the pattern $+0^{m-1}$ in the Aho–Corasick automaton). We have the following criterion for zero-capacity.

Theorem 3: The capacity of a set D is positive if and only if there is a path from q_{0^m} to $q_{+0^{m-1}}$ in the graph constructed above.

Proof: If $\mathrm{cup}(D) > 0$, by Theorem 2, there exists a word d, beginning with m zeros, and ending with $+0^{m-1}$, that avoids $D \cup -D$. Hence, entering this word in the automaton, the finite state will be (well defined and will be) the vertex labeled $+0^{m-1}$, because the vertices removed from the original automaton of Aho-Corasick do not make any problem, since we do not reach the vertices labeled with forbidden patterns. On the other hand, a path in the constructed graph represents an acceptable word, since it does not pass through any removed vertex, and hence no suffix of any prefix of this word will be in the forbidden set.

Moreover, a shortest path will give the shortest acceptable word, since the length of the path is equal to the length of the represented word. □

Corollary 5: The problem of determining whether or not the capacity of a given set of forbidden patterns is positive can be solved in polynomial time.

Proof: Aho shows in [1] that the automaton is constructible in polynomial time. The determination of the state q_{0^m} and the computation of the shortest path are obviously polynomially feasible. □

Corollary 6: If for a set D of forbidden patterns there are admissible words, then the length of a shortest admissible word does not exceed $2M + 2m$, where m is the maximal length of all patterns in D and M is the sum of the lengths of each forbidden pattern.

Proof: The number of vertices of the graph does not exceed $2M + m + 1$. Indeed, for each pattern of length l in $D \cup -D$ we add to the automaton at most l states, since there are no more than l prefixes of this pattern. We still add the pattern $\{+0^{m-1}\}$ (maximum m new states), and the root. If there is a path connecting two given vertices, this path can be chosen so that its length (in terms of number of vertices) will not exceed the total number of vertices (if it does not pass through the same vertex twice). Every edge of this path adds one bit to the admissible string. The initial length of the string is m (we start from 0^m), therefore the total length of the admissible word is at most $2M + 2m$. □

Proposition 2: If the capacity is positive, then $\mathrm{cap}(D) > 1/(2M + m)$, where m is the maximal length of all patterns in D and M is the sum of the lengths of each forbidden pattern.

Proof: If $\mathrm{cap}(D) > 0$, then there is an admissible string of length $n \leq 2M + 2m$ (Corollary 6). Consider the code given by (7). Its size is 2^l and the length of its words is at most

$$N_l = l(2M + 2m - m) = l(2M + m).$$

Therefore

$$
\begin{aligned}
\mathrm{cap}(D) &= \lim_{l \to \infty} \frac{\log_2 \delta_{N_l}}{N_l} \\
&\geq \lim_{l \to \infty} \frac{\log_2 2^l}{l(2M + m)} \\
&= \frac{1}{2M + m}.
\end{aligned}
$$ □

V. POSITIVE CAPACITY IS NP-HARD FOR EXTENDED SETS

We now consider the situation where forbidden patterns are allowed to contain the $\pm$ symbol. The symbol $\pm$ is to be understood in the following sense: whenever it occurs in a forbidden pattern, both the occurrences of $+$ and of $-$ are forbidden at that particular location. So, for example, avoiding the forbidden set $\{0 \pm + \pm\}$ is equivalent to avoiding the set $\{0 + ++, 0 + +-, 0 - ++, 0 - +-\}$. All results obtained for forbidden patterns over $\{-, 0, +\}$ have therefore their natural counterparts in the situation where the forbidden patterns are defined over the alphabet $\{-, 0, +, \pm\}$. In particular, the results of Section III do transfer *verbatim* and the bounds derived in Theorem 1 are valid exactly as stated there. We now prove a complexity result of capacity computation in this setup.

Theorem 4: The problem of determining if the capacity of a set of forbidden patterns over $\{0, +, -, \pm\}$ is equal to 0 is NP-hard.

Proof: The proof proceeds by reduction from the Not-All-Equal 3SAT problem that is known to be NP-complete (see [10]). In the Not-All-Equal 3SAT problem, we are given m binary variables $x_1, \ldots, x_m$ and n clauses that each contain three literals (a literal can be a variable or its negation), and we search a truth assignment for the variables such that each clause has at least one true literal and one false literal.

Suppose that we are given a set of clauses. We construct a set of forbidden patterns D such that $\mathrm{cap}(D) > 0$ if and only if the instance of Not-All-Equal 3SAT has a solution. The first part of D is given by

$$\{(0 \pm 0), (0 \pm \pm 0), \ldots, (0 \pm^{m-1} 0)\}. \tag{8}$$

Words over $\{-, 0, +\}$ that avoid these patterns are exactly those words for which any two consecutive zeros are either adjacent or have at least m symbols $+$ or $-$ between them. We use these m symbols as a way of encoding possible truth assignments for the variables.

We then add to D two patterns for every clause. These patterns are of length m and are entirely composed of symbols $\pm$, except for the positions corresponding to the three variables of the clause, which we set to $+$ if the clause contains the variable itself, or to $-$ if the clause contains the negation of the variable. We also add the opposite of this pattern; this last pattern is not necessary for the proof but preserves the symmetry of the construction.

For example, if the instance of Not-All-Equal 3SAT consists of the two clauses $(x_1, \bar{x}_3, x_4)$ and $(\bar{x}_2, x_4, x_5)$, the corresponding set D will be $D = \{(0 \pm 0), (0 \pm \pm 0), (0 \pm \pm \pm 0), (0 \pm \pm \pm \pm 0), (+ \pm - + \pm), (- \pm + - \pm), (\pm - \pm + +), (\pm + \pm - -)\}$. Such a set D has always a length polynomial in the number of clauses and the number of variables.

We now prove that there is a solution to the instance of Not-All-Equal 3SAT if and only if $\mathrm{cap}(D) > 0$. First, suppose that there exists a satisfying truth assignment for x and denote it by $\{\omega_1, \ldots, \omega_m\}$.

Associated to any $k \geq 1$ we construct a code of length $k(m+1)$ containing 2^k words as follows:

$$C_{k(m+1)} = \{0\omega 0\omega 0\omega 0 \dots 0\omega 0\omega, 0\omega 0\omega 0\omega 0 \dots 0\omega 0\bar{\omega},$$
$$0\omega 0\omega 0\omega 0 \dots 0\bar{\omega} 0\omega, \dots, 0\bar{\omega} 0\bar{\omega} 0\bar{\omega} 0 \dots 0\bar{\omega} 0\bar{\omega}\}$$

where $\omega = \omega_1 \dots \omega_m$.

Any difference between two words in this code is a word of the form $0z_1 0z_2 0 \dots 0z_k$ where for every $1 \leq i \leq k$, z_i is either a sequence of m 0's or a word of length m over $\{-, +\}$. Because ω satisfies the instance of Not-All-Equal 3SAT, these words avoid the set D constructed above. Moreover, the cardinality of $C_{k(m+1)}$ is 2^k and hence

$$\mathrm{cap}(D) \geq \lim_{k \to \infty} \log_2 2^{\frac{k}{k(m+1)}} = \frac{1}{m+1} > 0. \tag{9}$$

For the reverse implication, assume now that $\mathrm{cap}(D) > 0$. The capacity is positive, and so one can find two words whose differences contain a 0 and a $+$. But then since this difference must avoid the first part of the forbidden pattern, for a code C large enough, there must exist two words in the code whose difference contains a word over $\{-, +\}$ of length m. But this sequence avoids also the second part of D, and thus it represents an acceptable solution to our instance of Not-All-Equal 3SAT. $\square$

Note that a similar proof can be given if we replace the symbol "$\pm$" in the statement of the theorem by a symbol that represents either $+$, $-$, or 0.

VI. EXTREMAL NORMS AND COMPUTING THE CAPACITY

A classical way to estimate a joint spectral radius consists of computing successive upper bounds on it by applying the following well-known inequality:

$$\rho^n(\Sigma) \leq \max \{\|A_1 \dots A_n\| : A_i \in \Sigma\}$$

that holds for any norm, and any length n of the products. One could then hope that for a well-chosen norm, the joint spectral radius would be already obtained for $n = 1$, that is, for the set of matrices itself. This is the concept of extremal norm, which we now define properly. A norm $\|\cdot\|$ in $\mathbb{R}^d$ is called extremal for a family of operators $A_1, \dots, A_r$ if $\|A_i\| \leq \rho(\{A_1, \dots, A_r\})$ for all $i = 1, \dots, r$. The unit ball M of this norm is called the extremal convex body.

The notion *extremal* is justified by the fact that ρ is the smallest possible value such that the norms of all the operators $A_1, \dots, A_r$ do not exceed this value. The above inequality, which holds for any norm, for extremal norms becomes an equality (for all $n \geq 1$). If M is the unit ball corresponding to an extremal norm (an *extremal body*), $A_i M \subset \rho M$. On the other hand, any convex body M (convex compact with nonempty interior centrally symmetric with respect to the origin) possessing this property generates an extremal norm. It suffices to take the Minkowski norm defined by this body: $\|x\| = \inf \{\lambda > 0, \frac{1}{\lambda} x \in M\}$. Thus, there is a natural one-to-one correspondence between extremal norms and extremal bodies.

The existence of an extremal norm can simplify many problems related to the joint spectral radius, see [2], [16] and [18] for details. However, not every set of matrices possesses an extremal norm. The corresponding counterexamples are simple and well known. Sufficient conditions for the existence of an extremal norm can be found in [2], [16], [17]. For the matrices arising in the context of the capacity computation, however, these conditions are not always satisfied. Nevertheless, it turns out that in the case of capacity computation, the matrices do in fact always possess an extremal norm.

Theorem 5: For any set D of forbidden patterns the set $\Sigma(D)$ possesses an extremal norm.

Proof: Let $\Sigma(D)$ be the set of matrices corresponding to D. For a given point $x \geq 0$ let

$$\mathcal{O}(x) = \left\{\rho^{-n} A_1 \dots A_n x, A_i \in \Sigma(D), n \geq 0\right\}$$

be the normalized orbit of the point x under the action of all possible products of the operators in Σ. The product of length zero is defined as the identity operator, so the set $\mathcal{O}(x)$ contains x. Now define a set M as follows

$$M = \mathrm{Conv}\left\{\mathcal{O}(e_j), \mathcal{O}(-e_j), j = 1, \dots, 2^{m-1}\right\}$$

where $e_1, \dots, e_{2^{m-1}}$ are the canonical basis vectors in $\mathbb{R}^{2^{m-1}}$, and Conv denotes the convex hull. The set M is obviously convex, centrally symmetric with respect to the origin, and possesses a nonempty interior (because it contains the cross-polytope with the vertices $\{\pm e_j, j = 1, \dots, 2^{m-1}\}$ whose interior is nonempty). Moreover, Corollary 1 implies that the set $\mathcal{O}(x)$ is bounded for any x. To see this, replace ρ by $2^{\mathrm{cap}(D)}$ in Corollary 1 and recall that δ_n is, by definition of the set Σ, the maximal norm of products of length $n - (m-1)$ of matrices taken in Σ. Therefore M is bounded. So M is a convex body that possesses the property $A_i M \subset \rho M, i = 1, \dots, r$. Therefore it generates an extremal norm, and the theorem follows. $\square$

The very existence of an extremal norm for a set of matrices makes it possible to apply a geometric algorithm for computing a joint spectral radius with a given relative accuracy. We now briefly describe this algorithm; for all technical details we refer the reader to [16]. For the sake of simplicity, we consider the case of two matrices, the case of an arbitrary number of matrices is treated in the same way.

The algorithm. Suppose operators A_0, A_1 acting in $\mathbb{R}^d$ possess an extremal norm; one needs to find a number ρ^* such that $\frac{|\rho^* - \rho|}{\rho} < \varepsilon$, where $\varepsilon > 0$ is a given accuracy. Consider a sequence of convex polytopes $\{P_k\}$ produced as follows:

$$P_0 = \left\{(x_1, \dots, x_d) \in \mathbb{R}^d, \sum |x_i| \leq 1\right\}$$

is a cross-polytope. For any $k \geq 0$ the polytope P_{k+1} is an arbitrary polytope possessing the following properties. It is symmetric with respect to the origin, has at most $q(\varepsilon) = C_d \varepsilon^{\frac{1-d}{2}}$ vertices, where C_d is an effective constant depending only on d, and $(1 - \varepsilon)\bar{A} P_k \subset P_{k+1} \subset \bar{A} P_k$, where $\bar{A} X = \mathrm{Conv}\{A_0 X, A_1 X\}$.

After

$$N = \left\lceil \frac{3\sqrt{d} \ln \frac{c_2}{c_1}}{\varepsilon} \right\rceil$$

steps the algorithm terminates. The value $\rho^* = (v_{N+1})^{1/(N+1)}$ gives the desirable approximation of the joint spectral radius. Here v_k is the biggest distance from the origin to the vertices of the polytope P_k, c_1, c_2 are lower and upper bounds of the values $\{\rho^{-k} \max \{\|A_1 \dots A_k\|\}, k \in \mathbb{N}\}$. In our case the values c_1, c_2 can be taken directly from Theorem 1.

Each step requires us taking the convex hull of two polytopes having at most $q(\varepsilon)$ vertices and requires the approximation of one polytope with $2q(\varepsilon)$ vertices by a polytope with $q(\varepsilon)$ vertices with accuracy ε. Both operations are known to be polynomial w.r.t. $\frac{1}{\varepsilon}$ [16] (the dimension d is fixed). The computational complexity of this algorithm is $C \cdot \varepsilon^{-\frac{d+1}{2}}$, where C is some constant and $d = 2^{m-1}$. Therefore the algorithm is applicable for small values of m, say, for $m \leq 6$.

The complexity of this algorithm is exponential with respect to m, as the one proposed in Section III that approximates the capacity by successive estimations of δ_n. The advantages of one algorithm over the other appears in numerical computation of the capacity. In many

cases the approximation of invariant bodies by polytopes can lead to the exact value of the joint spectral radius. Suppose that by numerical observations we conjecture that ρ is attained by some product $\Pi_n = A_{i_1} \dots A_{i_n}$, i.e., $\rho = \rho(\Pi_n)^{1/n}$. If during the calculations we find a polytope P such that $\bar{A}P \subset \rho(\Pi_n)^{1/n}P$, then it occurs that $\rho = \rho(\Pi_n)^{1/n}$. As the polytope P we take $P = P_k = \mathrm{Conv}\{\bar{A}^j v - \bar{A}^j v, j = 0, \dots, k\}$ for some integer k, where v is the eigenvector of Π_n corresponding to the largest by modulo eigenvalue (we assume that this is real and unique).

Let us illustrate this method by computing the exact values of the capacity for several codes. In Examples 1 and 2, we find the values of capacities that were approximated in [15]. Example 3 deals with a code with $m = 4$.

Example 1:

$$\mathrm{cap}(\{0++\}) = \log_2 \rho(A_0) = \log_2\left(\frac{\sqrt{5}+1}{2}\right) = 0.69424191\dots$$

The eigenvector is $v = \left(2, \sqrt{5}-1, 2, \sqrt{5}-1\right)^T$. The algorithm terminates after five steps, the polytope $P = P_5$ has 32 vertices.

Example 2:

$$\mathrm{cap}(\{0+-\}) = \log_2 \rho(A_0) = \log_2\left(\frac{\sqrt{5}+1}{2}\right).$$

The algorithm terminates after four steps: $v = (2, \sqrt{5}-1, \sqrt{5}-1, 2)^T$, $P = P_4$, the polytope has 40 vertices.

Example 3:

$$\mathrm{cap}(\{+++-\}) = \log_2\left(\frac{\sqrt{3+2\sqrt{5}}+1}{2}\right)$$
$$= \log_2 \sqrt{\rho(A_0 A_1)} = 0.90053676\dots.$$

The algorithm terminates after eleven steps, the polytope $P = P_{11}$ has 528 vertices.

As illustrated in many applications it is quite often the case that the joint spectral radius is attained by some finite product. We say in these cases that the set of matrices possess the *finiteness property*. It was conjectured that all sets of matrices have the finiteness property: this is the well known finiteness conjecture which has been disproved in [7], [8], and [12]. Nevertheless, we conjecture here that the sets of matrices with binary entries, and, in particular, those constructed in order to compute a capacity do always possess the finiteness property. Numerical results in [11], [15], and in this paper seem to support this conjecture.

VII. CONCLUSION

One way to compute the capacity of a set of forbidden patterns is to compute the joint spectral radius of a set of matrices. In practice, this leads to a double difficulty: first, the size of the matrices is exponential in the size of the set of forbidden patterns, and second, the joint spectral radius is in general NP-hard to compute. We show in this paper that the simpler problem of checking the positivity of the capacity of a set defined on $\{+, -, 0\}$ is polynomially decidable but that the same problem becomes NP-hard when defined over the alphabet $\{+, -, 0, \pm\}$. We also provide bounds that allow faster computation of the capacity. Finally we prove the existence of extremal norms for the sets of matrices arising in the capacity computation and present a geometrical algorithm for capacity computation which we illustrate with several numerical examples. Even if this latter result allows to use algorithms that have proved to be quite efficient in practice, the

approach that consists of computing the joint spectral radius of the matrices defined in [15] cannot lead to a polynomial algorithm because of the exponential size of the sets of matrices.

REFERENCES

[1] A. V. Aho, "Algorithms for finding patterns in strings," in *Handbook of Theoretical Computer Science (vol. A): Algorithms and Complexity*. Cambridge, MA: MIT Press, 1990, pp. 255–300.

[2] N. Barabanov, "Lyapunov indicators of discrete inclusions. Part I, II and III," (in Russian) *Avtomat i Telemekh.*, vol. 2, pp. 40–46, 1988.

[3] M. A. Berger and Y. Wang, "Bounded semigroups of matrices," *J. Linear Algebra Appl.*, vol. 166, pp. 21–27, 1992.

[4] V. D. Blondel and Y. Nesterov, "Computationally efficient approximations of the joint spectral radius," *SIAM J. Matrix Anal.*, vol. 27, no. 1, pp. 256–272, 2005.

[5] V. D. Blondel and J. N. Tsitsiklis, "A survey of computational complexity results in systems and control," *Automatica*, vol. 36, no. 9, pp. 1249–1274, 2000.

[6] ——, "The boundedness of all products of a pair of matrices is undecidable," *Syst. Contr. Lett.*, vol. 41, no. 2, pp. 135–140, 2000.

[7] V. D. Blondel, J. Theys, and A. A. Vladimirov, "An elementary counterexample to the finiteness conjecture," *SIAM J. Matrix Anal.*, vol. 24, no. 4, pp. 963–970, 2003.

[8] T. Bousch and J. Mairesse, "Asymptotic height optimization for topical IFS, tetris heaps, and the finiteness conjecture," *J. Math. Amer. Soc.*, vol. 15, no. 1, pp. 77–111, 2002.

[9] I. Daubechies and J. C. Lagarias, "Sets of matrices all infinite products of which converge," *Linear Algebra Appl.*, vol. 161, pp. 227–263, 1992.

[10] M. R. Garey and D. S. Johnson, *Computers and Intractability—A Guide to the Theory of NP-Completeness*. San Francisco, CA: Freeman, 1979.

[11] R. Jungers, "NP-Completeness of the Computation of the Capacity of Constraints on Codes," Master's thesis, Université Catholique de Louvain-La-Neuve, Louvain-La-Neuve, Belgium, 2005.

[12] V. Kozyakin, "A dynamical systems construction of a counterexample to the finiteness conjecture," in *Proc. 44th IEEE Conf. Decision and Control (ECC 2005)*, 2005.

[13] B. E. Moision, A. Orlitsky, and P. H. Siegel, "On codes with local joint constraints," 2004, unpublished manuscript.

[14] B. E. Moision, A. Orlitsky, and P. H. Siegel, "Bounds on the rate of codes which forbid specified difference sequences," in *Proc. 1999 IEEE Global Telecommun. Conf. (GLOBECOM '99)*, Dec. 1999.

[15] ——, "On codes that avoid specified differences," *IEEE Trans. Inf. Theory*, vol. 47, no. 1, pp. 433–422, Jan. 2001.

[16] V. Protasov, "The joint spectral radius and invariant sets of linear operators," *Fundamentalnaya i Prikladnaya Matematika*, vol. 2, no. 1, pp. 205–231, 1996.

[17] ——, "On the asymptotics of the partition function," *Sb. Math.*, vol. 191, no. 3–4, pp. 381–414, 2000.

[18] ——, "The geometric approach for computing the joint spectral radius," in *Proc. 44th IEEE Conf. Decision and Control (ECC 2005)*, 2005.

[19] G. C. Rota and C. Strang, "A note on the joint spectral radius," *Indag. Math.*, vol. 22, pp. 379–381, 1960.

[20] J. Tsitsiklis and V. Blondel, "The Lyapunov exponent and joint spectral radius of pairs of matrices are hard—when not impossible—to compute and to approximate," *Math. Contr., Signals, Syst.*, vol. 10, no. 31, pp. 40–40, 1997.