

1 Recognizable Picture Languages: Separating UREC 2 from coUREC via Communication Complexity

3 Antonin Callard ✉

4 LIRMM – CNRS and Université de Montpellier, France

5 Andrei Romashchenko ✉

6 LIRMM – CNRS and Université de Montpellier, France

7 Véronique Terrier ✉

8 Université Caen Normandie, ENSICAEN, CNRS, Normandie Univ, GREYC; F-14000 Caen, France

9 Pascal Vanier ✉

10 Université Caen Normandie, ENSICAEN, CNRS, Normandie Univ, GREYC; F-14000 Caen, France

11 — Abstract —

12 We introduce communication-complexity lifting techniques into the study of recognizable picture
13 languages. As an application, we resolve a long-standing open problem of Anselmo et al. (2006) by
14 constructing a language in UREC whose complement does not belong to REC. Our proof of the
15 lower bound is inspired by the communication-complexity approach that Göös et al. (2022) used for
16 unambiguous automata, although the setting of picture languages requires substantially different
17 technical tools.

18 **2012 ACM Subject Classification** Theory of computation → Models of computation; Theory of
19 computation → Formal languages and automata theory; Mathematics of computing → Discrete
20 mathematics

21 **Keywords and phrases** picture languages, two-dimensional languages, unambiguous tiling systems,
22 communication complexity

1 Introduction

In this work, we study nondeterminism in the theory of recognizable picture languages. In particular, we consider the implications of restricting nondeterminism to unambiguous nondeterminism.

Generally speaking, nondeterminism can be seen as a form of exhaustive search in which several computational branches are considered in parallel. A fundamental question in theoretical computer science is whether such a parallel exhaustive search can be replaced by an efficient deterministic computation. This question appears in many models of computation and has been widely studied in complexity theory.

Besides the total number of possible computational branches, another important parameter when talking about nondeterminism is the number of *accepting* branches. Restricting this number to at most one branch leads to the notion of unambiguous computation and to the complexity class UP, which is a natural refinement of NP. The importance of unambiguous computations was illustrated, for example, by the famous Valiant–Vazirani theorem [37], which gives a randomized reduction from NP to problems with a unique witness.

Similar questions appear in formal languages and automata theory (see [14] for a survey). The classical equivalence between deterministic and nondeterministic finite automata shows that nondeterminism does not increase the expressive power of finite automata, but may significantly reduce their number of states. The difference between general nondeterministic and unambiguous nondeterministic finite automata is a classical direction of research, see, e.g., [27, 33, 34, 24]. Recently, the power of unambiguous nondeterminism in finite automata has again received attention, in particular in [31, 39, 16].

In this paper, we study similar questions in the context of picture languages. More precisely, the class of *recognizable* picture languages is a natural two-dimensional analogue of regular languages of finite words. In this setting, the classes REC and UREC respectively refer to the nondeterministic and unambiguously nondeterministic recognizable picture languages, and are thus similar to (word) languages recognized by nondeterministic and unambiguous finite automata. Although the class REC and its variants have been studied extensively, the structural consequences of restricting nondeterminism to the unambiguous case with the UREC class are still not well understood. Our goal is to study the relationship between the classes REC/coREC and UREC/coUREC and to clarify the role of unambiguity in the theory of two-dimensional words/pictures.

1.1 Recognizable picture languages: the formal definitions

We now recall the formal definitions of recognizable picture languages and their unambiguous variants, as introduced by Giammarresi & Restivo in [10]. This definition is based upon a notion of locally valid tilings.

► **Definition 1** (picture). A picture of size $m \times n$ over an alphabet Σ is a rectangular pattern $p \in \Sigma^{m \times n}$. A picture language is a subset of Σ^{**} , where Σ^{**} is the set of all pictures over Σ .

Let $\#$ be a special symbol not in Σ . The bordered picture of p is the picture $\hat{p}$ of size $(m+2) \times (n+2)$ obtained by surrounding the picture p with $\#$ symbols.

► **Definition 2** (patches, valid picture). A patch is a picture of size 2×2 . Given an alphabet Σ and a set of patches Θ over $\Sigma \cup \{\#\}$, a picture p is Θ -valid if all the subpictures of size 2×2 in the bordered picture $\hat{p}$ belong to Θ .

A language that can be described by the set of its 2×2 -subpictures is called *local*:

67 ► **Definition 3** (local picture language). *A picture language L over Σ is local if there exists a*
 68 *set of patches Θ over $\Sigma \cup \{\#\}$ such that L is exactly the set of Θ -valid pictures. In this case*
 69 *we write $L = \text{Local}(\Theta)$.*

70 Among their many characterizations, regular languages of finite words are exactly the
 71 morphic images of local languages. Giammarresi and Restivo extended this notion to picture
 72 languages by introducing in [10] the class REC of pictures languages *recognized* as projections
 73 of local tiling systems:

74 ► **Definition 4** (projection). *For Γ and Σ two alphabets and $\pi : \Gamma \rightarrow \Sigma$ a map, the projection*
 75 *of a picture $p \in \Gamma^{m \times n}$ is the picture $\pi(p) \in \Sigma^{m \times n}$ given by $\pi(p)(i, j) = \pi(p(i, j))$.*
 76 *The projection $\pi(L)$ of a language L over Γ is the language over Σ defined as $\{\pi(p) : p \in L\}$.*

77 ► **Definition 5** (tiling system). *A tiling system is a quadruple $\mathcal{T} = (\Sigma, \Gamma, \Theta, \pi)$ where*
 78 *— Σ and Γ are finite alphabets;*
 79 *— Θ is a set of patches over $\Gamma \cup \{\#\}$;*
 80 *— $\pi : \Gamma \rightarrow \Sigma$ is an alphabetic map.*
 81 *The language L recognized by the tiling system $\mathcal{T}$ is the projection of the local language*
 82 *$\text{Local}(\Theta)$, that is, $L = \pi(\text{Local}(\Theta))$. A language is set to be recognizable if it is recognized*
 83 *by some tiling system $\mathcal{T}$.*

84 ► **Definition 6** (REC). *The family of recognizable languages is denoted by REC, and the*
 85 *family of complements of recognizable languages by coREC.*

86 ► **Remark 7.** The choice of sizes 2×2 for patches in this serie of definitions is rather arbitrary.
 87 Equivalently, REC can be defined as projections of languages specified by local constraints
 88 in terms of *dominos* (i.e., patches of size 2×1 and 1×2), see [11].

89 A tiling system is called unambiguous if every picture has a unique preimage in its
 90 corresponding local language.

91 ► **Definition 8** (UREC). *A tiling system $\mathcal{T} = (\Sigma, \Gamma, \Theta, \pi)$ recognizing a language L is*
 92 *unambiguous if every picture $p \in L$ admits a unique preimage $q \in \text{Local}(\Theta)$ such that $p = \pi(q)$.*
 93 *A picture language that is recognized by an unambiguous tiling system is called unambiguous.*
 94 *UREC denotes the family of all unambiguous recognizable picture languages, and coUREC*
 95 *the family of their complements.*

96 1.2 Brief historical survey

97 Although various two-dimensional automata and grammars had been studied much earlier, it
 98 was in the 1990s that *recognizable picture languages* emerged as a coherent research area.
 99 In the 1990s, Giammarresi and Restivo introduced the class REC and laid the foundations
 100 of the theory of recognizable picture languages [10, 11]. Inoue and Takanami connected
 101 this framework with two-dimensional automata [20]. Giammarresi, Restivo, Seibert, and
 102 Thomas provided a logical characterization of recognizability in terms of monadic second-
 103 order logic [13]. Thus, the same class received equivalent descriptions in terms of local tilings,
 104 automata, and logic. Latteux and Simplot established connections between recognizable
 105 picture languages and classical one-dimensional context-sensitive languages [23], and Matz
 106 developed logical and algebraic methods for obtaining negative results, in particular for
 107 proving that certain picture languages are not recognizable [28].

108 In the 2000s, Anselmo, Giammarresi, Madonia, and Restivo studied determinism and
 109 ambiguity in REC [3]. Their further results concerned deterministic and unambiguous

subclasses of REC, as well as necessary conditions for recognizability [2, 4]. We omit several other results that are less directly related to the questions considered in this paper.

These works established REC as a natural two-dimensional counterpart of the class of regular languages. However, they also revealed that the relationships between recognizability, determinism, ambiguity, and complementation are more subtle for picture languages than for word languages [11, 3].

1.3 Major structural questions

Concerning the relationships between the classes REC, coREC, UREC, and coUREC, let us recall two important facts:

- REC is not closed under complement, *i.e.*, $\text{REC} \neq \text{coREC}$; see [35];
 - there exist inherently ambiguous picture languages in REC, *i.e.*, $\text{UREC} \subsetneq \text{REC}$; see [3].
- These results left open two major questions.

► **Question 1** (1994, [29, 12]). *Is there a language in $(\text{REC} \cap \text{coREC}) \setminus (\text{UREC} \cup \text{coUREC})$?*

► **Question 2** (2006, [3]). *Is UREC closed under complement? Equivalently, is there a picture language in $\text{UREC} \setminus \text{coUREC}$?*

Question 1 was answered positively by Véronique Terrier in 2019 [36]. In this paper, we answer Question 2 by showing that UREC is not closed under complement. In fact, we prove the stronger statement:

► **Theorem 9.** *There is a picture language in $\text{UREC} \setminus \text{coREC}$.*

Similarly to [36], which answered Question 1, our proof of Theorem 9 uses methods from communication complexity: we realize a communication problem with a suitable communication complexity as a specifically constructed picture language. We discuss this technique in more detail in Section 3.

More specifically, for the proof of Theorem 9, we use a communication problem with the following property: the unambiguous communication complexity of computing a certain predicate should be *rather low*, while the communication complexity of computing the negation of this predicate should be *rather high* (in the proof, the words “rather high” and “rather low” will get a precise meaning). It turns out that classical methods of communication complexity, in particular the matrix rank method, do not allow us to achieve this goal. The issue is that the ranks of a Boolean matrix and of its complement cannot differ significantly. Therefore, to estimate the communication complexity, we use a different approach — the lifting technique introduced in [32, 18].

Lower bounds for communication complexity obtained by lifting method have been used, for similar reasons, in the study of unambiguous finite automata, see [16]. However, our application of this method involves a technical feature¹ which, to the best of our knowledge, has not appeared in previous applications of the method.

The rest of the paper is organized as follows. In Section 2, we present the necessary tools from communication complexity, including the lifting technique. In Section 3.1, we

¹ The lifting technique provides a transformation of certificate problem for a function $f(x_1, \dots, x_n)$ into a communication complexity problem. This transformation involves a *gadget* (a function with particular properties of a randomness extractor; in practice one can take the inner product). In usual applications of the lifting technique, the length of the gadget’s input is logarithmic in n ; in our construction, we take a gadget with much longer inputs, see Remarks 23 and 26 and in Section 3.1, and Section 2.2 for a detailed discussion.

148 describe the construction of a language in $\text{UREC} \setminus \text{coUREC}$. In Section 3.2, we prove that the
 149 constructed language does not belong to coUREC by using techniques from communication
 150 complexity. In Section 3.3 we prove that the constructed language belongs to UREC .

151 2 Technical tools

152 We first introduce two notions of complexity for Boolean functions. The *certificate com-*
 153 *plexity* measures how many bits a Boolean function needs to read from its input; and the
 154 *communication complexity* measures the amount of communication needed to compute a
 155 Boolean function whose input is distributed between two (or more) participants. These two
 156 notions of complexity are related: the *lifting technique* mentioned above is a method that
 157 transforms a certificate problem into a communication problem of similar complexity.

158 2.1 Certificate complexity

159 In the classical *decision tree* model of computation (see, *e.g.*, [8]), the measure of complexity
 160 of a Boolean function in n variables $f(x_1, \dots, x_n)$ is the minimum number of input variables
 161 that need to be assigned a value in order to definitely establish the value of f . This notion is
 162 closely connected with the idea of *certificate*, which is a partial assignment of Boolean values
 163 to the variables $(x_1, \dots, x_n)$ that determines the value of a Boolean function.

164 ► **Example 10.** For the function called *majority* of n Boolean arguments

$$165 \quad \text{majority}_n(x_1, \dots, x_n) = \begin{cases} 1, & \text{if } \geq n/2 \text{ values } x_i \text{ are equal to } 1, \\ 0, & \text{if } < n/2 \text{ values } x_i \text{ are equal to } 1 \end{cases}$$

166 in case $n = 7$, a partial assignment $(1, 0, 1, *, 1, 1, *)$ implies that the value of *majority* is
 167 equal to 1, whatever values we substitute on the place of $*$'s. Thus, this partial assignment
 168 “certifies” that $\text{majority}_7(x_1, \dots, x_7)$ is equal to 1.

169 The *complexity* of a certificate is the number of assigned variables: in the previous example,
 170 the complexity of $(1, 0, 1, *, 1, 1, *)$ is equal to 5. Furthermore, the *certificate complexity*
 171 of a function f is the minimal integer k such that one can find a family of certificates of
 172 complexity $\leq k$ covering exactly the assignments on which f evaluates to true².

173 Let us proceed with formal definitions. We say that a *clause* is a finite conjunction of
 174 literals, where a literal is either a propositional variable or its negation. The *width* of a
 175 clause is the number of literals it contains. Every clause of width k can be understood as a
 176 representation of a Boolean function (in variables $x_1, \dots, x_n$, where $n \geq k$).

177 A *DNF formula* is a disjunction of such clauses. Let $D = C_1 \vee \dots \vee C_m$ be an n -variable
 178 DNF formula. The *width* of this DNF formula is the maximum of the widths of the clauses C_i
 179 for $i = 1, \dots, m$. This DNF is called *unambiguous* if, for every input $(x_1, \dots, x_n) \in \{0, 1\}^n$,
 180 at most one of the clauses C_i evaluates to true, *i.e.* $C_i(x_1, \dots, x_n) = 1$.

² Technically, the formal definition that we discuss in this section corresponds to the complexity of *nondeterministic* decision trees.

The certificate complexities are then defined as follows:

► **Definition 11.** For any Boolean function $f : \{0,1\}^n \rightarrow \{0,1\}$, the nondeterministic certificate complexity $N^{cert}(f)$ is the smallest integer k such that f can be written as a DNF of width k ; and its unambiguous certificate complexity $UN^{cert}(f)$ is the smallest integer k such that f can be written as an unambiguous DNF of width k .

Besides, we write $coN^{cert}(f)$ for $N^{cert}(\neg f)$ and $coUN^{cert}(f)$ for $UN^{cert}(\neg f)$.

► **Example 12.** For the disjunction function

$$OR_n(x_1, \dots, x_n) = \begin{cases} 1, & \text{if } x_i = 1 \text{ for at least one } i \\ 0, & \text{otherwise} \end{cases}$$

the definition implies $N^{cert}(OR_n) = 1$ (it is enough to guess one variable x_i such that $x_i=1$) and $UN^{cert}(OR_n) = n$ (we have to ask the values of all variables).

We refer the reader to [8, 1, 7] for a more detailed discussion of the definition of certificate complexity and further examples.

We finally recall a family of results from Göös et al. proving that unambiguous certificate complexity and co-certificate complexity can be uniformly bounded away from one another:

► **Theorem 13** ([15, 6]). There exist real constants $\alpha < \beta$, an infinite family of indices $\mathcal{K} \subset \mathbb{N}$ and some associated functions $f_k : \{0,1\}^k \rightarrow \{0,1\}$ for $k \in \mathcal{K}$ such that $UN^{cert}(f_k) \leq O(k^\alpha)$ and $coN^{cert}(f_k) \geq \Omega(k^\beta)$. Moreover, the family $\{f_k\}$ is explicit: there exists a deterministic algorithm that, given k and $x \in \{0,1\}^k$, computes $f_k(x)$ in time $\text{poly}(k)$.

► **Remark 14.** From the proof of Theorem 13 it follows that the set $\mathcal{K}$ is polynomial-time computable and, moreover, there is an explicitly given polynomial-time computable function $t(k)$ that is an upper bound for $UN^{cert}(f_k)$ and a lower bound for $coN^{cert}(f_k)$, i.e., such that we have $UN^{cert}(f_k) \ll t(k) \ll coN^{cert}(f_k)$ for $k \in \mathcal{K}$.

► **Remark 15.** Different concrete instantiations of this theorem (with different constructions of f_k and different parameters α, β) are given in a series of works [15, 17, 18, 6]. In [15], one obtains a separation with some constant $\alpha > 1$ via a complicated recursive construction. In [6], one achieves a near-quadratic separation, i.e., β/α can be made arbitrarily close to 2.

2.2 Communication complexity

The theory of communication complexity studies how much information two (or more) parties need to exchange in order to compute together a function when each party is initially given only part of the input. There are several settings in which the parties act deterministically, nondeterministically, or use randomness; for comprehensive surveys, we refer to [22, 30].

In this paper, we study the *nondeterministic* and *unambiguous nondeterministic* versions of communication complexity as defined below:

► **Definition 16** (protocols). For a given function $f : X \times Y \rightarrow \{0,1\}$ a nondeterministic communication protocol of cost k is a pair of functions $A : X \times \{0,1\}^k \rightarrow \{0,1\}$ and $B : Y \times \{0,1\}^k \rightarrow \{0,1\}$ such that: for all $(x,y) \in X \times Y$, we have $f(x,y) = 1$ if and only if there exists some certificate $z \in \{0,1\}^k$ satisfying $A(x,z) = 1$ and $B(y,z) = 1$. Such a protocol is called *unambiguous* if for each tuple $(x,y) \in X \times Y$ such that $f(x,y) = 1$ there is exactly one z such that $A(x,z) = 1$ and $B(y,z) = 1$.

Informally, a nondeterministic communication protocol can be understood as a game of three parties: Alice (who is given an input $x \in X$) and Bob (who is given an input $y \in Y$), who aim at checking if $f(x, y) = 1$; and an omniscient but untrustworthy referee (who can access both inputs x and y). To proceed, the referee broadcasts a k -bits message z , which is meant to be a certificate proving $f(x, y) = 1$. If Alice and Bob confirm that z is compatible with their respectively input parts x and y , one can be sure that indeed $f(x, y) = 1$. The *cost* of the protocol is the length of the referee's message (we do not count Alice's and Bob's confirmations).

For every f there is a trivial protocol: the referee broadcasts the entire pair (x, y) , so that Alice and Bob learn each other's part of the input, and this is obviously enough to compute $f(x, y)$. However, some functions f admit much more efficient protocols, like in the following example:

► **Example 17.** Denote $\text{NEQ}_n : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ the non-equality predicate

$$\text{NEQ}_n(x_1 \dots x_n, y_1 \dots y_n) = \begin{cases} 1, & \text{if there exists } i \text{ such that } x_i \neq y_i, \\ 0, & \text{if } x_i = y_i \text{ for all } i. \end{cases}$$

Let us define the following protocol: the referee announces the binary expansion of an index $i \in \{1, \dots, n\}$ and a Boolean value b , which is meant to be a position i such that $x_i \neq y_i$, and $b = x_i$ is the value of Alice's input at the i -th position. If Alice confirms that $x_i = b$ and Bob confirms that $y_i \neq b$, then Alice's and Bob's input parts indeed differ.

The cost of this protocol is $\lceil \log n \rceil + 1$. However, it is *not* unambiguous: if x and y differ in several positions, the referee can broadcast several indices certifying that $x \neq y$.

► **Remark 18.** Let $f : X \times Y \rightarrow \{0, 1\}$ be a Boolean function. Denote by M^f the matrix with $|X|$ rows and $|Y|$ columns such that $M_{x,y}^f = f(x, y)$ for all $x \in X$ and $y \in Y$ (the rows and columns of the matrix are indexed by elements from X and Y respectively). Let $A \subset X$ and $B \subset Y$ be such that $M_{x,y}^f = 1$ for all $(x, y) \in A \times B$. Then the minor of M^f formed by the intersection of rows from A and columns from B is called a *combinatorial 1-rectangle*.

Observe that nondeterministic communication protocol of cost k for f induces a *cover* of all 1-entries of M^f by at most 2^k combinatorial 1-rectangles. Similarly, an unambiguous nondeterministic communication protocol of cost k induces a *partition* of the 1-entries of M^f into at most 2^k disjoint combinatorial 1-rectangles.

► **Definition 19** (communication complexity). Let $f : X \times Y \rightarrow \{0, 1\}$ be a function with binary values. The nondeterministic communication complexity of this function is the minimal cost k over all nondeterministic protocols for f . We denote it $\text{N}^{\text{cc}}(f)$.

The unambiguous (nondeterministic) communication complexity of f is the minimal cost k over all unambiguous nondeterministic protocols for f . We denote it $\text{UN}^{\text{cc}}(f)$. We use the notation $\text{coN}^{\text{cc}}(f)$ and $\text{coUN}^{\text{cc}}(f)$ for $\text{N}^{\text{cc}}(\neg f)$ and $\text{UN}^{\text{cc}}(\neg f)$ respectively.

► **Remark 20.** Equivalently, the nondeterministic communication complexity of f can be defined as logarithm of the *cover number*, i.e., of the minimum size of a cover of the 1-entries of M^f with combinatorial 1-rectangles. Similarly, the unambiguous nondeterministic communication complexity of f can be defined as logarithm of the *partition number*, i.e., of the minimum size of a partition of the 1-entries of M^f into disjoint combinatorial 1-rectangles.

► **Example 21.** Denote $\text{EQ}_n : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ the equality predicate defined as

$$\text{EQ}_n(x, y) = \begin{cases} 1, & \text{if } x = y, \\ 0, & \text{otherwise.} \end{cases}$$

For this Boolean function $\text{N}^{\text{cc}}(\text{EQ}_n) = n$ and $\text{coN}^{\text{cc}}(\text{EQ}_n) = \log n + 1$, see, e.g., [22].

263 The lifting method with the inner-product gadget from [15, theorem 4] and [17] (see
 264 also [16, theorem 7]) allows to transfer certificate complexity results in the domain of
 265 communication complexity:

266 ► **Theorem 22** (Lifting with the inner-products). *Let $f : \{0, 1\}^k \rightarrow \{0, 1\}$ be any Boolean*
 267 *function and let $b = b(k)$ satisfy $b = \Omega(\log k)$, with a sufficiently large multiplicative constant.*
 268 *We define $g : \{0, 1\}^b \times \{0, 1\}^b \rightarrow \{0, 1\}$ by $g(u, v) = \langle u, v \rangle \bmod 2$.*
 269 *Define $F_k = f \circ g^k$ on inputs $(x, y) \in (\{0, 1\}^b)^k \times (\{0, 1\}^b)^k$ by*

$$270 \quad F_k(x, y) = f(g(x^{(1)}, y^{(1)}), \dots, g(x^{(k)}, y^{(k)})), \quad (1)$$

271 where each $x^{(j)}$ and $y^{(j)}$ is an array of b input variables. Then

$$272 \quad \text{UN}^{cc}(F) \leq O(\text{UN}^{cert}(f) \cdot b), \quad (2)$$

273 and

$$274 \quad \text{coN}^{cc}(F) \geq \Omega(\text{coN}^{cert}(f) \cdot b). \quad (3)$$

275 The upper bound (2) comes from an explicit protocol: given a fixed unambiguous DNF φ
 276 for f of minimal width, the referee chooses one clause from φ (which is supposed to certify
 277 that the value of f is *True*), and broadcasts the chosen clause together with a valuation of
 278 its variables and the corresponding pairs $(x^{(j)}, y^{(j)})$; Alice and Bob then confirm that the
 279 broadcasted clause is indeed a clause in φ , that the broadcasted parts of their input are
 280 correct, and that the broadcasted valuation corresponds to the inner products $g(x^{(j)}, y^{(j)})$.

281 The lower bound (3) is a much more involved result; it follows from [17, Theorem 2].

282 ► **Remark 23.** In the literature this theorem is usually formulated and applied with $b = \Theta(\log k)$
 283 but the proof works for all b bigger than $\Omega(\log k)$. We will later use this theorem for some
 284 $b(k)$ that is much larger than k .

285 Combined with Theorem 13, we deduce:

286 ► **Corollary 24** (Communication separation for lifted functions). *Let $\mathcal{K} \subset \mathbb{N}$ be a set of natural*
 287 *numbers, and let $\{f_k\}_{k \in \mathcal{K}}$ be a family of Boolean functions $f_k : \{0, 1\}^k \rightarrow \{0, 1\}$ such that*
 288 *for some $\alpha < \beta$, we have $\text{UN}^{cert}(f_k) \leq O(k^\alpha)$ and $\text{coN}^{cert}(f_k) \geq \Omega(k^\beta)$.*

289 *Let $b = b(k)$ satisfy $b = \Omega(\log k)$, and let $g : \{0, 1\}^b \times \{0, 1\}^b \rightarrow \{0, 1\}$ be the inner-product*
 290 *gadget $g(u, v) = \langle u, v \rangle \bmod 2$. Define $F_k := f_k \circ g^k$ as in Theorem 22, with one-party input*
 291 *length $n := k \cdot b(k)$; then $\text{UN}^{cc}(F_k) \leq O(k^\alpha \cdot b(k))$ and $\text{coN}^{cc}(F_k) \geq \Omega(k^\beta \cdot b(k))$.*

292 ► **Remark 25.** In the well-known work [15], a separation between UN^{cc} and coN^{cc} was
 293 established for the very natural problem CLIQUE vs. INDEPENDENT SET. Unfortunately,
 294 this problem has logarithmic UN^{cc} . This is too low for our applications, so we cannot base
 295 our proof on this result.

296 ► **Remark 26.** The smaller the parameter $b = b(k)$, the bigger the relation between the
 297 lower and the upper bound in Corollary 24. In the field of communication complexity (see
 298 [15, 17, 16]), the authors employ this machinery with the smallest possible $b(k)$ (that is
 299 $b(k) = \lambda \log k$ for some large enough constant λ). We, on the contrary, will rely on a very
 300 large $b(k)$ to prove Theorem 9.

3 Proof of the main result

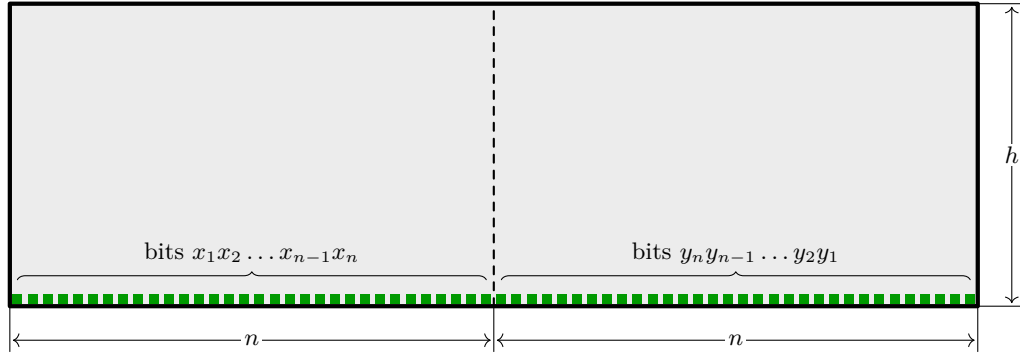
3.1 Construction

In order to prove Theorem 9, we will create a picture language from the communication problem given by Corollary 24. First of all, we define a simple scheme that assigns picture languages to families of Boolean functions.

► **Definition 27.** For a family $\mathbf{F} = \{F_k\}_{k \in \mathcal{K}}$ of functions $F_k: \{0, 1\}^{n(k)} \times \{0, 1\}^{n(k)} \rightarrow \{0, 1\}$, and a function $h: \mathbb{N} \rightarrow \mathbb{N}$, we define the picture language $L_{\mathbf{F}, h}$ over the alphabet $\mathcal{A} = \{0, 1, \square\}$ composed of patterns w on $2n(k)$ columns and $h(k)$ rows such that (see Figure 1):

- the bottom row of w consists of $2 \cdot n(k)$ cells that can be either 0 or 1; these cells represent two bit strings $x_1 \dots x_{n(k)}$ and $y_1 \dots y_{n(k)}$ such that $F_k(x_1 \dots, x_{n(k)}, y_1 \dots y_{n(k)}) = 1$; we assume that the bits $y_1 \dots y_{n(k)}$ are presented in the picture in reverse order, see Fig. 1.
- all cells of the other $h - 1$ rows of the patterns are blank symbols $\square$.

If we consider the communication problem in which Alice and Bob are each given a pattern of size $n(k) \times h(k)$, and want to check if the concatenation of these patterns forms a picture in $L_{\mathbf{F}, h}$, we essentially re-obtain the communication problem given by the family of functions $\mathbf{F}$. Indeed, it reduces to giving Alice and Bob the bottom rows of their patterns, and computing the value of the corresponding Boolean function F_k on these inputs.



■ **Figure 1** Blueprint of a pattern of size $2n \times h$ for $n := n(k)$ and $h := h(k)$. The two blocks of green dots represent Alice's bits $\mathbf{x} = x_1 \dots x_n$ and Bob's $\mathbf{y} = y_1 \dots y_n$. The bits of Bob's input are presented in reverse order. The pattern belongs to the picture language $L_{\mathbf{F}, h}$ iff $F_k(\mathbf{x}, \mathbf{y}) = 1$.

Fixing principal parameters of the construction

In this section, we fix the family $\mathbf{F} = \{F_k\}_{k \in \mathcal{K}}$ and the function $h: \mathbb{N} \rightarrow \mathbb{N}$ as follows:

- We set an explicit family $\mathbf{f} = \{f_k\}_{k \in \mathcal{K}}$ given by Theorem 13, i.e. whose complexity satisfies $\text{UN}^{\text{cert}}(f_k) = O(k^\alpha)$ and $\text{coN}^{\text{cert}}(f_k) = \Omega(k^\beta)$ for some $\alpha < \beta$.
- We fix $b(k)$ such that $b(k) = 2^{2^{\Theta(k)}}$ and $k \mapsto k \cdot b(k)$ is injective, and define $n(k) := k \cdot b(k)$.
- We define the family $\mathbf{F} = \{F_k\}_{k \in \mathcal{K}}$ as $F_k = f_k \circ g^k$, where g the inner-product gadget on bit-strings of length $b(k)$.
- Finally, we fix a function $h(k) := \lfloor k^\gamma \cdot b(k) \rfloor$ such that $\text{UN}^{\text{cc}}(F_k) \ll h(k) \ll \text{coN}^{\text{cc}}(F_k)$.

Without loss of generality we will assume that $b(k)$ and $h(k)$ are fully time-constructible functions³.

By interpreting the patterns of the resulting language $L_{\mathbf{F},h}$ as instances of a communication problem $F_k: \{0,1\}^{n(k)} \times \{0,1\}^{n(k)} \rightarrow \{0,1\}$, we obtain the following results:

- In Section 3.2 we prove that $L_{\mathbf{F},h}$ is not in coREC. This is the main contribution of the paper. The proof is based upon the fact that $N^{cc}(\neg F_k) \gg h(k)$, so that the frontier between Alice's and Bob's halves of the patterns (the dashed line in Figure 1) is too small to encode communication transcripts.
- In Section 3.3 we provide a proof that $L_{\mathbf{F},h}$ is in UREC. The argument is based on the fact that $UN^{cc}(F_k) \leq h(k)$: we construct a tiling system that “geometrically” simulates a specific communication protocol for $\{F_k\}_{k \in \mathcal{K}}$ between Alice and Bob.

3.2 Proof that $L_{\mathbf{F},h}$ cannot be in coREC

► **Lemma 28.** *Let $\mathbf{F} = \{F_k\}_{k \in \mathcal{K}}$ be a family of Boolean functions*

$$F_k: \{0,1\}^{n(k)} \times \{0,1\}^{n(k)} \rightarrow \{0,1\},$$

and $h: \mathbb{N} \rightarrow \mathbb{N}$ be a function on natural numbers. Let $L_{\mathbf{F},h}$ be the picture language defined for these F_k and h , as explained in Section 3.1. If $L_{\mathbf{F},h}$ is in REC, then the nondeterministic communication complexity of F_k is at most $O(h(k))$.

► **Remark 29.** Such a condition was first observed in [5]. It was also an important ingredient of the argument in [36].

Proof. Assume that a picture language $L_{\mathbf{F},h}$ admits a tiling system (in the sense of Definition 6): there exists a local picture language L over some finite alphabet Σ and a letter projection $\pi: \Sigma \rightarrow \{0,1,\square\}$ such that any rectangular pattern P belongs to $L_{\mathbf{F},h}$ if and only if there exists a pattern $Q \in L$ such that P is a letter-wise π -projection of Q .

For $(x,y) \in \{0,1\}^n \times \{0,1\}^n$ a pair of strings, we denote by $P_{x,y}$ the pattern of size $(2n) \times h(n)$ as shown in Figure 1 (the colors in the bottom row represent the bits of x and y , and all other cells are blank). We define a communication protocol as follows:

- The referee chooses a picture Q in L such that $\pi(Q) = P_{x,y}$; and broadcasts the subpicture Z of size $2 \times (h+2)$ obtained as the two middle columns of the bordered picture $\hat{Q}$;
- Alice accepts the referee's message if there exists a (partially bordered) picture $\hat{Q}_L$ of size $(n+2) \times (h+2)$ such that:
 - $\hat{Q}_L$ is compatible with Z , which means that the two *rightmost* columns of $\hat{Q}_L$ coincide with Z , as shown in Figure 2;
 - The picture $\hat{Q}_L$ satisfies the local constraints associated with the local language L ;
 - The π -projection of the non-bordered Q_L corresponds with Alice's input x ;
- Similarly, Bob accepts the referee's message if there exists a (partially bordered) picture $\hat{Q}_R$ of size $(n+2) \times (h+2)$ such that:
 - $\hat{Q}_R$ is compatible with Z , which means that the two *leftmost* columns of $\hat{Q}_R$ coincide with Z , as shown in Figure 2;

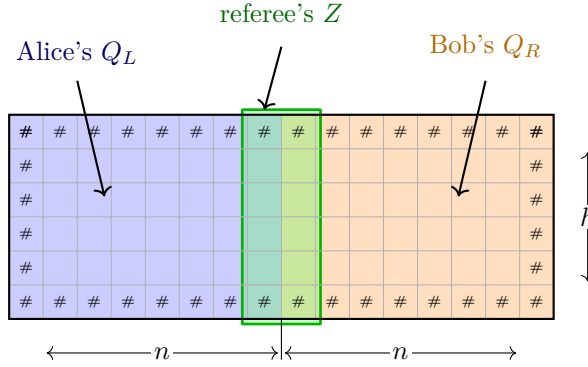
³ A function $f: \mathbb{N} \rightarrow \mathbb{N}$ is called fully time-constructible if there exists a Turing machine that on every input of length n runs for exactly $f(n)$ steps, see, e.g., [19]. If a function $f: \mathbb{N} \rightarrow \mathbb{N}$ is computable in time $O(f(n))$ and satisfies $f(n) \geq (1+\varepsilon)n$ for some $\varepsilon > 0$ and all sufficiently large n , then it is fully time-constructible [21]. In particular, most commonly used superlinear functions, such as $f(n) = n^k$ for $k = 2, 3, \dots$, $f(n) = 2^n$, or $f(n) = 2^{2^n}$, are fully time-constructible.

- 364 ■ The picture $\hat{Q}_R$ satisfies the local constraints associated with the local language L ;
- 365 ■ The π -projection of the non-bordered Q_R corresponds with Bob's input y (in reverse);
- 366 Observe that the patterns $\hat{Q}_L$ and $\hat{Q}_R$ compatible with the broadcasted Z may be not unique,
- 367 and may not coincide with the bordered picture $\hat{Q}$ chosen by the referee.

368 By construction, the communication complexity of the protocol is $O(h)$, since the referee's
 369 message consists of $2(h+2)$ letters from a finite alphabet Σ . We now prove that this defines
 370 a valid protocol for F_k .

371 If $F_k(x, y) = 1$, then this protocol accepts the pair (x, y) . Indeed $P_{x,y} \in L_{\mathbf{F},h}$, so there
 372 exists a valid pattern $Q \in L$ such that $P_{x,y} = \pi(Q)$. The referee can broadcast the two
 373 central columns of $\hat{Q}$; and Alice and Bob can use the left and right halves of $\hat{Q}$ as $\hat{Q}_L$ and $\hat{Q}_R$,
 374 which are locally valid and coincide with their respective inputs x and y (see Figure 2).

375 Conversely, we prove that if a pair (x, y) is accepted by the protocol, then $F_k(x, y) = 1$.
 376 Indeed, if a broadcasted pattern Z is accepted by Alice and Bob, then this Z can be extended
 377 into a locally valid bordered picture $\hat{Q}$ such that $\pi(Q) = P_{x,y}$: indeed, there then there exist
 378 two $(n+2) \times h$ partially bordered patterns $\hat{Q}_L$ and $\hat{Q}_R$ extending Z to the left and the right
 379 respectively; furthermore, their combination $\hat{Q}$ into a $(2n+2) \times (h+2)$ bordered picture is
 380 locally valid; so that the pattern Q actually satisfies $Q \in L$, and then $P_{x,y} = \pi(Q) \in L_{\mathbf{F},h}$.
 381 By definition of $L_{\mathbf{F},h}$, we conclude that $F_k(x, y) = 1$. ◀



■ **Figure 2** The patterns from the proof of Lemma 28.

382 Lemma 28 implies immediately the following corollary, which means that $L_{\mathbf{F},h}$ constructed
 383 in Section 3.1, is not in coREC. This concludes the proof of our main result.

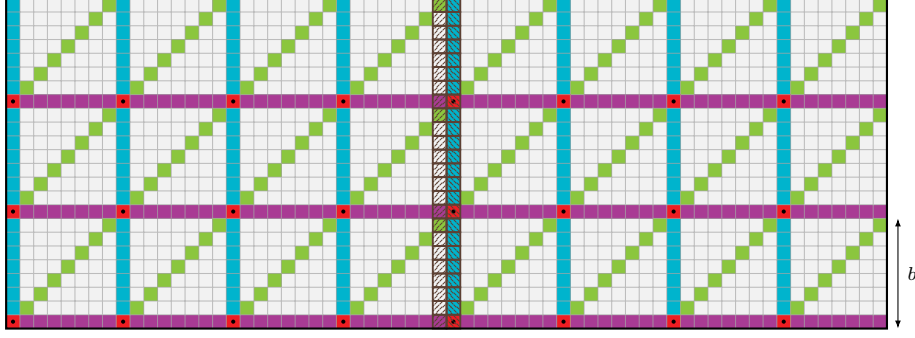
384 ► **Corollary 30.** Let $\mathbf{F} = \{F_k\}_{k \in \mathcal{K}}$ be a family of Boolean functions

$$385 \quad F_k : \{0, 1\}^{n(k)} \times \{0, 1\}^{n(k)} \rightarrow \{0, 1\},$$

386 and $h : \mathbb{N} \rightarrow \mathbb{N}$ be a function on natural numbers, and let $L_{\mathbf{F},h}$ be the picture language from
 387 Definition 27. If $\text{coN}^{cc}(F_k) \gg h(k)$, then the picture language $L_{\mathbf{F},h}$ is not in coREC.

388 3.3 Proof that $L_{\mathbf{F},h}$ is in UREC

389 In this section we design a local picture language L over some alphabet $\mathcal{A}$ and define a map
 390 $\pi : \mathcal{A} \rightarrow \{0, 1, \square\}$ such that the π -projections of pictures in L are exactly the pictures in $L_{\mathbf{F},h}$;
 391 and every picture of $L_{\mathbf{F},h}$ has a unique preimage in L . This will prove that $L_{\mathbf{F},h} \in \text{UREC}$.



■ **Figure 3** Schematics of a valid picture in L : Layer 1 draws square grids and Layer 2 selects a pair of adjacent vertical strips.

In Layer 1, vertical ■ lines and horizontal ■ form a grid. Diagonal ■ lines running from southwest to northeast enforce that each cell of this grid is a square, and that all squares are of the same size; as we enforce with the local rules that whenever two of these three types of lines (horizontal, vertical, diagonal) meet, the third type must pass through the same point ■. We also enforce in the local rules that the boundary of the picture aligns with the grid.

In Layer 2, two adjacent lines of ■ and ■ symbols select a pair of vertical strips called the *Middle Line*. The local rules require that the line of ■ symbols aligns with a vertical ■ line. However, Layer 2 does not uniquely determine the position of the Middle Line: additional constraints will be added to force the Middle Line to lie at the horizontal center of the picture, with the same number of columns on either side.

While the construction is a little technical, its individual ingredients are elementary and this proof should be routinely reproducible by researchers in multidimensional symbolic dynamics. We therefore only give a concise and schematic description of the construction.

At a high level, the construction simulates in L the communication protocol for the proof of $\text{UN}^{cc}(F) \leq O(\text{UN}^{cert}(f) \cdot b)$ in Theorem 22, which certifies the true values of the Boolean functions $F_k = f_k \circ g^k$. As is often the case in the construction of complex picture languages, we will describe the language L as a superimposition of *layers*. That is, the alphabet of L will be a Cartesian product $\mathcal{A} = \mathcal{A}_0 \times \mathcal{A}_1 \times \dots \times \mathcal{A}_7$, and we will refer to the projection onto the component $\mathcal{A}_i$ of a picture $Q \in L$ as the i^{th} layer of Q . This conceptual separation of pictures into layers allows for more an intuitive description of the language L , as most local constraints in L (though not all) will enforce a prescribed structure one layer at a time.

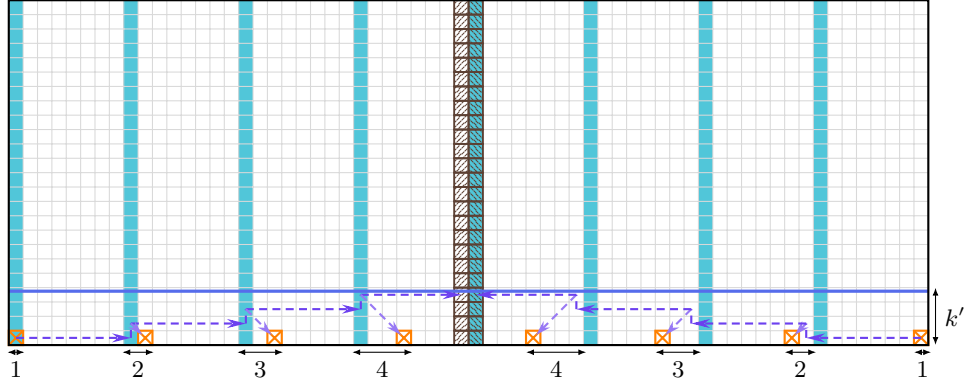
Layer 0: Base Layer. We set $\mathcal{A}_0 = \{0, 1, \square\}$ and denote $\pi : \mathcal{A} \rightarrow \{0, 1, \square\}$ the restriction onto the $\mathcal{A}_0$ -component. This layer contains the picture from $L_{\mathbf{F}, h}$ that we wish to certify.

Layer 1: Grid Layer. We do not specify the alphabet $\mathcal{A}_1$ explicitly, but instead explain the geometric patterns implemented on this layer. Layer 1 draws a regular grid of square-shaped cells with vertical ■ lines and horizontal ■, and a system of diagonal ■ lines enforce the regularity and the squared shape of the grid cells. See Figure 3 for more details.

In the rest of this proof, and for any valid picture $Q \in L$, we denote b the grid spacing of the 1st Layer of Q . At this stage, the spacing b is not fixed: its value will be determined below through constraints coupling Layer 1 to the other layers.

Layer 2: Middle Line. On this layer, we mark two adjacent columns with ■ and ■ tiles (shown in brown in Figure 3) and refer to them collectively as *the Middle Line*. While one can enforce with local rules (*e.g.* colored area) that there is exactly one Middle Line, its position is not fixed at this stage: the middle position will be enforced by Layer 3.

Layer 3: Block Counting. The grid from Layer 1 (superimposed on the bottom row of Layer 0) partitions the bottom row of Layer 0 into consecutive *blocks of b bits*. We denote these blocks by $x^{(1)}, x^{(2)}, \dots$ (counting from left to right) if they appear on the left of the Middle Line, and by $y^{(1)}, y^{(2)}, \dots$ (counting from right to left) if they appear on its right.



■ **Figure 4** Schematics of a valid picture in L : Layer 3 draws a system of $\boxtimes$ unary markers, each marking the index of a block $x^{(i)}$ and $y^{(i)}$.

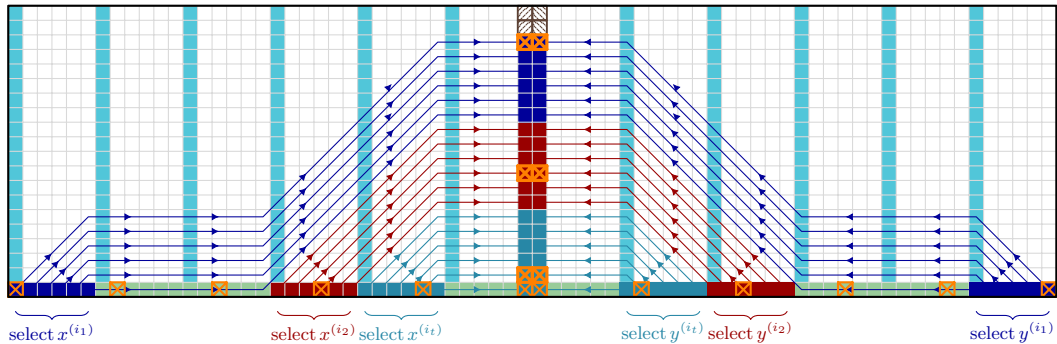
In the left half of the picture, the dashed lines represent a signal propagating from left to right, moving one step upward each time it crosses a vertical blue line, and then being reflected toward the bottom row to place the corresponding $\boxtimes$ unary marker. These markers serve as indices of the b -bit blocks. In the right half of the picture the construction is symmetric.

By requiring that the counters of the left and right halves have the same height when entering the Middle Line, we ensure that the Middle Line indeed bisects the picture. We draw a horizontal bold blue line at this maximal height k' . We will use this horizontal line once more when discussing Layer 7.

420 In *Layer 3*, we introduce a geometric structure assigning an *index* to each such block.
 421 The indices are represented by simple unary counters (using $\boxtimes$ symbols) and enumerate the
 422 blocks from left to right, starting at 1. Thus, every b -bit block in the bottom row of Layer 0
 423 is assigned an index. See Figure 4 for more details.

424 This construction naturally provides us with another useful feature: we can ensure that
 425 the two halves of a valid pattern have the same number of b -bit blocks, and hence that the
 426 Middle Line indeed bisects the picture. In the rest of this proof, we denote k' the maximal
 427 value of the counter, which corresponds to the height of the bold blue line in Figure 4.

428 *Layer 4: Block Selection.* On this layer, some of the b -bit blocks are marked as *selected*.
 429 The bits of each selected block, together with its index, are transmitted by a system of “wires”
 430 towards the vertical Middle Line. The local rules ensure that blocks with the same indices
 431 have to be selected on the two sides of the Middle Line. Thus, in a valid picture of the
 432 language L , there exists some indices $i_1, \dots, i_t$ such that the selected blocks are $x^{(i_1)}, \dots, x^{(i_t)}$
 433 on the left, and $y^{(i_1)}, \dots, y^{(i_t)}$ on the right. For more details, see Figure 5.

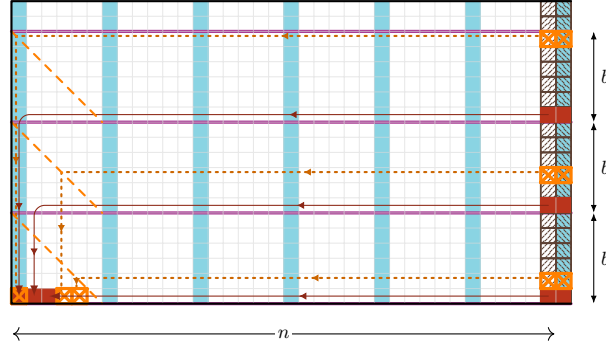


■ **Figure 5** Schematics of a valid picture in L : Layer 4 selects blocks of bits $x^{(i_j)}$ and $y^{(i_j)}$ and wires them towards the vertical Middle Line.

On the left half of a picture, we implement “wires” heading either right or diagonally until they arrive to the target vertical line. The value of the selected input bits (from Layer 0) together with the unary counter markers $\boxtimes$ (from Layer 3) on the bottom row spread along the corresponding wires. The symmetric scheme is used to transmit the selected $y^{(i_j)}$ on the right half of the picture.

434 *Layer 5: Inner Products.* This layer processes the bits of the selected blocks from Layer 4
 435 appearing on the Middle Line: the local rules implement a running-sum computation of the
 436 inner products over $\mathbb{F}_2$ for each selected pair $(x^{(i_j)}, y^{(i_j)})$. The corresponding inner-product
 437 values are “recorded” along the Middle Line, at the bottom of each block.

438 *Layer 6: Certificate Transmission.* This layer again implements signal propagation along
 439 geometric structures that plays the role of “communication wires”, as shown in Figure 6, to
 440 collect the following data in a compact form drawn at the bottom-left corner of the picture:
 441 (i) the indices of the bit blocks selected by Layer 4 as a string `select_ind` $\in \{\square, \boxtimes\}^*$.
 442 (ii) the inner-product values computed in Layer 5 for the corresponding pairs of selected
 443 blocks, in the form of a string `inner_prod` $\in \{0, 1\}^t$.



■ **Figure 6** Schematics of a valid picture in L : Layer 6 wires the indices of the selected blocks $(x^{(i_j)}, y^{(i_j)})$ and the values of the corresponding inner products (as they appear on Layer 5) to a compact form appearing on the bottom-left corner of the picture.

The result bits of the inner products, highlighted as ■ in the figure, are moved horizontally and then downwards. The unary markers $\boxtimes$ (corresponding to the selected indices) are moved horizontally (dotted lines) until they cross a dashed diagonal, and then downwards, thus forming a bitmap of size k containing $\boxtimes$ at the selected positions and blanks otherwise.

444 *Layer 7: Computation Layer.* This layer represents a space-time diagram of a Turing
 445 machine. This machine operates on finitely many *tapes*, and each tape has some moving
 446 *heads* that read and write symbols on the tapes’ cells (we may allow several heads in one
 447 tape). We specify below the input supplied to this machine and the computation it performs.

448 We embed the space-time diagram of a Turing machine. The *space* is horizontal: every
 449 row of the picture represents a Turing machine configuration that includes the data written
 450 on the tapes, the positions of the heads, and the internal state of the machine. The *time*
 451 is vertical and goes from the bottom to the top. The coherence of such a diagram can be
 452 easily implemented as a local picture language, see *e.g.*, [25, 38, 26]. We assume that the
 453 computation starts with all heads on the first cell (the leftmost tile of the row). In the local
 454 rules, we also enforce that valid pictures only embed successfully terminating computations.

455 We now specify the 9 tapes of our multitape Turing machine and their roles:

- 456 ■ one read-only tape contains the unary representation of an integer denoted by k ; its value
 457 is guessed nondeterministically in bottom row of the diagram and is kept intact;
- 458 ■ one tape takes as an input the gap between the two leftmost vertical ■ lines of the grid
 459 of Layer 1 (which can be interpreted as the parameter b in the unary expansion);
- 460 ■ two other tapes take as their inputs the bit strings `select_ind` and `select_ind`, which
 461 are directly copied from Layer 6 (where they already appear on the bottom-left corner);
- 462 ■ one tape uses input k to count from 1 to $b(k)$ (recall that $b(k)$ is a fully time-constructible
 463 function) and makes sure that $b = b(k)$ matches as specified in the definition of $L_{F,h}$;

- 464 ■ another tape uses input k to count from 1 to $h(k)$ (recall that $h(k)$ is also a fully time-constructible function) and makes sure that this run is over exactly at the top row for the picture (the last row of the space-time diagram);
- 465
- 466
- 467 ■ one more tape is used to check that the number of b -bit blocks encoded in both halves of the picture (*i.e.* the number of blocks $x^{(i)}$ and $y^{(j)}$) is equal to k ; to this end, we check that the height k' of the bold $-$ line on Layer 3 is equal to k (kept on the first tape);
- 468
- 469
- 470 ■ the last tape uses the compact certificates `select_ind` and `inner_prod` to certify that the function $F_k = f \circ g^k$ from Theorem 22 is equal to 1 on the bit strings embedded into the bottom row of the picture.
- 471
- 472

473 Let us provide some more details on this last item:

- 474 1. Using the last working tape, the machine first checks that k actually belongs to the set of integers $\mathcal{K}$ indexing the family $\mathbf{f} = \{f_k\}_{k \in \mathcal{K}}$ fixed in Section 3.1.
- 475
- 476 2. The machine then enumerates the list of all DNF formulas on k variables, and selects from this enumeration the unambiguous DNF φ of minimal cost representing the Boolean function f_k (if there are several such DNF, it takes the first in lexicographic order). This search is much larger than k (there are $2^{2^{O(k)}}$ such formulas) but small compared to $b(k)$.
- 477
- 478
- 479
- 480 3. Finally, the machine checks that the indices encoded in `select_ind` correspond exactly to the list of variables in one of the clauses in φ , and the partial valuation assigning the truth values from `inner_prod` to these variables evaluations the clause to true.
- 481
- 482

483 If any of these checks fails, the machine should stop and reject the input; otherwise, it accepts. Since the latter computation only depends on k , we can assume that the computation implemented on the last tape halts in time $\ll h(k)$. Therefore, we have enough room to embed the entire space-time diagrams in the pictures of size $h \times (2n)$.

487 *End of the proof.* It remains to verify that the constructed layers together provide an unambiguous local language for $L_{\mathbf{F},h}$.

489 **Proof.** For every picture $P_{x,y} \in L_{\mathbf{F},h}$, we prove that there exists some picture $Q \in L$ such that $\pi(Q) = P$. Indeed, let $k \in \mathcal{K}$ be the parameter corresponding to the picture P (recall that the width function $n: k \mapsto n(k)$ is injective, so k is uniquely defined). We define Layer 0 of Q by embedding the bits of $P_{x,y}$, and then define the content of Layers 1 to 7:

- 493 1. Layer 1: we draw a grid of $2k$ columns and k rows of $b(k) \times b(k)$ square cells;
- 494
- 495 2. Layer 2: we position the Middle Line in the middle of the picture;
- 496
- 497 3. Layer 3 is entirely determined by Layer 1;
- 498
- 499 4. Layer 4: We select pairs of blocks $(x^{(i_1)}, y^{(i_1)}), \dots, (x^{(i_t)}, y^{(i_t)})$ with the indices $(i_1, \dots, i_t)$ taken from the certificate ensuring $f_k(z_1, \dots, z_k) = 1$, where $z_j = g(x^{(j)}, y^{(j)})$, $j = 1, \dots, k$, are the values of the inner product of the selected blocks);
- 500
- 501 5. Layers 5 and 6 are then entirely determined;
- 502
- 503 6. Layer 7: we write the integer k on the first tape of the Turing machine. All the computations of the machine are then deterministic.

504 Thus, we obtain a picture Q that satisfies the local constraints of L and such that $\pi(Q) = P$.

505 It remains to observe that such a Q is uniquely determined since any other choices in the layers of Q would eventually lead to a contradiction with the local constraints of L : for example, Layer 3 enforces the position of the Middle Line on Layer 2, and its bold $-$ line constraints the nondeterministic choice of k on Layer 7. The most interesting is the non-determinism from the block selection on Layer 4: it is actually entirely determined because the “canonical” DNF φ representing f_k that is computed on Layer 7 is unambiguous; given (x, y) such that $F_k(x, y) = 1$, there exists exactly one possible block selection $(x^{(i_j)}, y^{(i_j)})_{1 \leq j \leq t}$.

506 This concludes the proof. $\blacktriangleleft$

4 Conclusion

The main proof presented in this paper designs a picture language embedding a well-chosen communication problem, and is consequently somewhat technical. The proof involves several geometric gadgets and computation embeddings of Turing machine computations via their space-time diagrams. There remains an interesting open problem to find a more natural example of a language in $\text{UREC} \setminus \text{coREC}$, with a simpler proof of membership in UREC . Another natural problem to consider would be whether UREC is closed under union.

It would also be interesting to investigate the extent to which the method developed in this paper can be adapted to study the properties of sofic shifts over multidimensional lattices (*e.g.* Weiss' question about the entropies of their finite type covers [9]).

References

- 1 Andris Ambainis, Martins Kokainis, and Robin Kothari. Nearly optimal separations between communication (or query) complexity and partitions. In *31st Conference on Computational Complexity, CCC 2016, Tokyo, Japan, May 29 - June 1, 2016*, volume 50 of *LIPIcs*, pages 4:1–4:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016. URL: <https://doi.org/10.4230/LIPIcs.CCC.2016.4>.
- 2 Marcella Anselmo, Dora Giammarresi, and Maria Madonia. Deterministic and unambiguous families within recognizable two-dimensional languages. *Fundamenta Informaticae*, 98(2-3):143–166, 2010. doi:10.3233/fi-2010-221.
- 3 Marcella Anselmo, Dora Giammarresi, Maria Madonia, and Antonio Restivo. Unambiguous recognizable two-dimensional languages. *RAIRO-Theoretical Informatics and Applications*, 40(2):277–293, 2006. doi:10.1051/ita:2006008.
- 4 Marcella Anselmo and Maria Madonia. Classes of two-dimensional languages and recognizability conditions. *RAIRO-Theoretical Informatics and Applications*, 44(4):471–488, 2010. doi:10.1051/ita/2011003.
- 5 Marcella Anselmo and Maria Madonia. A stronger recognizability condition for two-dimensional languages. *Discrete Mathematics & Theoretical Computer Science*, 15(Automata, Logic and Semantics), 2013. doi:10.46298/dmtcs.599.
- 6 Kaspars Balodis, Shalev Ben-David, Mika Göös, Siddhartha Jain, and Robin Kothari. Unambiguous DNFs and Alon-Saks-Seymour. *SIAM Journal on Computing*, pages 121–157, 2023. doi:10.1137/22M1480616.
- 7 Shalev Ben-David, Pooya Hatami, and Avishay Tal. Low-sensitivity functions from unambiguous certificates. In *8th Innovations in Theoretical Computer Science Conference (ITCS 2017)*, pages 28–51, 2017. doi:10.4230/LIPIcs.ITCS.2017.28.
- 8 Harry Buhrman and Ronald De Wolf. Complexity measures and decision tree complexity: a survey. *Theoretical Computer Science*, 288(1):21–43, 2002. doi:10.1016/S0304-3975(01)00144-X.
- 9 Angela Desai. Subsystem entropy for $\mathbb{Z}^d$ sofic shifts. *Indagationes Mathematicae*, 17(3):353–359, 2006. doi:10.1016/S0019-3577(06)80037-6.
- 10 Dora Giammarresi and Antonio Restivo. Recognizable picture languages. *International Journal of Pattern Recognition and Artificial Intelligence*, 6(2&3):241–256, 1992. doi:10.1142/S021800149200014X.
- 11 Dora Giammarresi and Antonio Restivo. Two-dimensional languages. In *Handbook of formal languages: volume 3 beyond words*, pages 215–267. Springer, 1997. doi:10.1007/978-3-642-59126-6_4.
- 12 Dora Giammarresi and Antonio Restivo. Matrix-based complexity functions and recognizable picture languages. In *Logic and Automata: History and Perspectives [in Honor of Wolfgang Thomas]*, pages 307–330, 2008.

- 559 13 Dora Giammarresi, Antonio Restivo, Sebastian Seibert, and Wolfgang Thomas. Monadic
560 second-order logic over rectangular pictures and recognizability by tiling systems. *Information*
561 *and computation*, 125(1):32–45, 1996. doi:10.1006/inco.1996.0018.
- 562 14 Jonathan Goldstine, Martin Kappes, Chandra MR Kintala, Hing Leung, Andreas Malcher,
563 and Detlef Wotschke. Descriptive complexity of machines with limited resources. *Journal of*
564 *Universal Computer Science*, 8(2):193–234, 2002. doi:10.3217/jucs-008-02-0193.
- 565 15 Mika Göös. Lower bounds for clique vs. independent set. In *2015 IEEE 56th Annual Symposium*
566 *on Foundations of Computer Science*, pages 1066–1076. IEEE, 2015. doi:10.1109/FOCS.2015.
567 69.
- 568 16 Mika Göös, Stefan Kiefer, and Weiqiang Yuan. Lower bounds for unambiguous automata
569 via communication complexity. In *ICALP 2022*, volume 229, pages 126:1–126:13, 2022.
570 doi:10.4230/LIPIcs.ICALP.2022.126.
- 571 17 Mika Göös, Shachar Lovett, Raghu Meka, Thomas Watson, and David Zuckerman. Rectangles
572 are nonnegative juntas. *SIAM Journal on Computing*, 45(5):1835–1869, 2016. doi:10.1137/
573 15M103145X.
- 574 18 Mika Göös, Toniann Pitassi, and Thomas Watson. Deterministic communication vs. partition
575 number. *SIAM Journal on Computing*, 47(6):2435–2450, 2018. doi:10.1137/16M1059369.
- 576 19 Steven Homer and Alan L. Selman. *Computability and complexity theory*, volume 194. Springer,
577 2001. doi:10.1007/978-1-4614-0682-2.
- 578 20 Katsushi Inoue and Itsuo Takanami. A characterization of recognizable picture languages.
579 *International journal of pattern recognition and artificial intelligence*, 8(02):501–508, 1994.
580 doi:10.1007/3-540-56346-6_35.
- 581 21 Kojiro Kobayashi. On proving time constructibility of functions. *Theoretical Computer Science*,
582 35:215–225, 1985. doi:10.1016/0304-3975(85)90015-5.
- 583 22 Eyal Kushilevitz and Noam Nisan. *Communication complexity*. Cambridge University Press,
584 1997. doi:10.1017/CB09780511574948.
- 585 23 Michel Latteux and David Simplot. Context-sensitive string languages and recognizable picture
586 languages. *Information and Computation*, 138(2):160–169, 1997. doi:10.1006/inco.1997.
587 2659.
- 588 24 Hing Leung. Structurally unambiguous finite automata. In *International Conference on*
589 *Implementation and Application of Automata*, pages 198–207. Springer, 2006. doi:10.1007/
590 11812128_19.
- 591 25 Leonid A. Levin. Fundamentals of computing (a cheatlist). *ACM SIGACT News*, 27(3):89,
592 1996. doi:10.1145/235666.235680.
- 593 26 Harry R. Lewis and Christos H. Papadimitriou. Elements of the theory of computation. *ACM*
594 *SIGACT News*, 29(3):62–78, 1998.
- 595 27 Oleg B. Lupanov. A comparison of two types of finite automata. *Problemy kibernetiki*,
596 9:321–326, 1963. doi:10.1007/3-540-18740-5.
- 597 28 Oliver Matz. On piecewise testable, starfree, and recognizable picture languages. In *Inter-*
598 *national Conference on Foundations of Software Science and Computation Structure*, pages
599 203–210. Springer, 1998. doi:10.1007/BFb0053551.
- 600 29 Andreas Pothhoff, Sebastian Seibert, and Wolfgang Thomas. Nondeterminism versus deter-
601 minism of finite automata over directed acyclic graphs. *Bulletin of the Belgian Mathematical*
602 *Society-Simon Stevin*, 1(2):285–298, 1994. doi:10.36045/bbms/1103408550.
- 603 30 Anup Rao and Amir Yehudayoff. *Communication complexity: and applications*. Cambridge
604 University Press, 2020. doi:10.1017/9781108671644.
- 605 31 Mikhail A. Raskin. A superpolynomial lower bound for the size of non-deterministic complement
606 of an unambiguous automaton. In *45th International Colloquium on Automata, Languages,*
607 *and Programming, ICALP 2018, Prague, Czech Republic, July 9-13, 2018*, volume 107 of
608 *LIPIcs*, pages 138:1–138:11. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. doi:
609 10.4230/LIPIcs.ICALP.2018.138.

- 610 32 Ran Raz and Pierre McKenzie. Separation of the monotone NC hierarchy. *Combinatorica*,
611 19(3):403–435, 1999. doi:10.1007/s004930050062.
- 612 33 Erik Meineche Schmidt. Succinctness of descriptions of context-free, regular, and finite
613 languages. *DAIMI Report Series*, 6(84), 1978.
- 614 34 Richard Edwin Stearns and Harry B. Hunt III. On the equivalence and containment problems
615 for unambiguous regular expressions, regular grammars and finite automata. *SIAM Journal*
616 *on Computing*, 14(3):598–611, 1985. doi:10.1137/0214044.
- 617 35 Andrzej Szepietowski. Two-dimensional on-line tessellation acceptors are not closed under
618 complement. *Information Sciences*, 64(1-2):115–120, 1992. doi:10.1016/0020-0255(92)
619 90114-N.
- 620 36 Véronique Terrier. Communication complexity tools on recognizable picture languages. *Theo-*
621 *retical Computer Science*, 795:194–203, 2019. doi:10.1016/j.tcs.2019.05.040.
- 622 37 Leslie Valiant and Vijay Vazirani. NP is as easy as detecting unique solutions. *Theoretical*
623 *Computer Science*, 47(3):85–93, 1986. doi:10.1145/22145.22196.
- 624 38 Peter van Emde Boas et al. The convenience of tilings. In *Complexity, logic, and recursion*
625 *theory*, Lecture Notes in Pure and Applied Mathematics, pages 331–363. Dekker, New York,
626 1997.
- 627 39 Cas Widdershoven. *Unambiguous and finitely ambiguous automata*. PhD thesis, University of
628 Oxford, 2022.