High Complexity Tilings with Sparse Errors *

Bruno Durand[1], Andrei Romashchenko[2], Alexander Shen[3]

[1] LIF Marseille, CNRS & Univ. Aix–Marseille, Bruno.Durand@lif.univ-mrs.fr
[2] LIF Marseille, CNRS & Univ. Aix–Marseille, on leave from IITP RAS, andrei.romashchenko@gmail.com
[3] LIF Marseille, CNRS & Univ. Aix–Marseille, on leave from IITP RAS, alexander.shen@lif.univ-mrs.fr

Abstract. Tile sets and tilings of the plane appear in many topics ranging from logic (the Entscheidungsproblem) to physics (quasicrystals). The idea is to enforce some global properties (of the entire tiling) by means of local rules (for neighbor tiles). A fundamental question: Can local rules enforce a complex (highly irregular) structure of a tiling?

The minimal (and weak) notion of irregularity is *aperiodicity*. R. Berger constructed a tile set such that every tiling is aperiodic. Though Berger's tilings are not periodic, they are very regular in an intuitive sense.

In [3] a stronger result was proven: There exists a tile set such that all $n \times n$ squares in all tilings have Kolmogorov complexity $\Omega(n)$, i.e., contain $\Omega(n)$ bits of information. Such a tiling cannot be periodic or even computable.

In the present paper we apply the fixed-point argument from [5] to give a new construction of a tile set that enforces high Kolmogorov complexity tilings (thus providing an alternative proof of the results of [3]). The new construction is quite flexible, and we use it to prove a much stronger result: there exists a tile set such that all tilings have high Kolmogorov complexity even if (sparse enough) tiling errors are allowed.

1 Introduction

Tiles are unit squares with colored sides. We may place translated copies of the same tile into different cells of a cell paper (rotations are not allowed). Tiles in the neighbor cells should match (common side must have the same color in both).

More formally, we consider a finite set C of *colors*. A *tile* is a quadruple of colors (left, right, top and bottom ones), i.e., an element of C^4. A *tile set* is a subset $\tau \subset C^4$. A *tiling* of the plane with tiles from τ (τ-*tiling*) is a mapping $U\colon \mathbb{Z}^2 \to \tau$ that respects the color matching condition.

It is well known that local rules can enforce some kind of irregularity in tilings. This first and simplest example of *irregularity property* is aperiodicity. The following classical result was proven by Berger [1]:

* Supported by NAFIT ANR-08-EMER-008-01 and RFBR 09-01-00709-a grants

Theorem 1. *There exists a tile set τ such that τ-tilings exist and all of them are not periodic.* [1]

Berger's tilings are aperiodic, but their structure is very regular and rather simple to describe. A tile set which enforces irregularity in a much stronger sense was constructed in [3]: it is a tile set that accepts only tilings of high Kolmogorov complexity. As a tiling is an infinite object, we look at finite patterns in a tiling and measure their Kolmogorov complexity:

Theorem 2. *There exists a tile set τ such that τ-tilings exist and all $n \times n$ squares in all τ-tilings have Kolmogorov complexity $\Omega(n)$.*

More precisely the result can be reformulated as follows: there exists a tile set τ and constants c_1 and c_2 such that τ-tilings exist and in every τ-tiling every $n \times n$-square has Kolmogorov complexity at least $c_1 n - c_2$. The lower bound $\Omega(n)$ in this theorem is tight: if τ-tilings exist, in one of them every $n \times n$ square has complexity $O(n)$ (see the discussion in [3]).

We stress that it is more reasonable to investigate the minimal complexity of a τ-tiling (for a given tile set τ), not its maximal complexity. Indeed, the maximal complexity can be very large (of order $O(n^2)$) for a trivial tile set. E.g., for the set τ of *all* tiles with black and white edges, there is a τ-tiling that contains every $n \times n$ pattern, and maximal complexity is $\Omega(n^2)$.

We refer to [3] for a more detailed discussion and philosophical motivation of Theorem 2. In this paper we give a proof of Theorem 2 and generalize it for tilings *with sparse random errors*.

The precise statement of our result about tilings with random errors requires some technical definitions, see Section 4. Here we explain the intuitive idea. We consider 'faulty' tilings of the plane, where the local tiling rules are not true for the entire plane (as we required in the usual definition of tiling) but can be violated on a sparse randomly chosen set of cells. (This generalization of the standard definition of tiling looks rather natural: in physical crystal grids we usually expect that local rules can be violated at sparse points.) Then we construct such a tile set that even 'faulty' tilings (for *almost all* sets of faults) must have high Kolmogorov complexity.

The paper is organized as follows. In Section 2 we remind the fixed-point construction of an aperiodic tile set [5] that is used as a starting point.

In Section 3 we provide a new proof of Theorem 2 using fixed point construction with variable zoom factors. The new proof is simpler in some respects then the original one from [3], and can be generalized to the case when sparse errors are allowed.

Finally, in Section 4 we briefly explain our most difficult result: a "robust" tile set such that all tilings, even with a sparsely placed errors, have linear complexity of fragments. To achieve this result, we combine several ideas: (1) a fixed-point tile set with variable zoom factors; (2) calculation and propagation of Reed–Solomon's checksums, and (3) covering of a random sparse set by a hierarchical family of isolated islands (technically, we need to update the construction from

[5, 14] and use bi-islands instead of simple islands). For the lack of space, the technical details are omitted in the conference version of the paper.

2 Fixed-point aperiodic tile set

In this section we remind the fixed point construction of aperiodic tile sets from [5] (the reader familiar with the arguments from [5] can safely skip this part).

2.1 Macro-tiles

Fix a tile set τ and an integer $N > 1$ (*zoom factor*). A *macro-tile* is an $N \times N$ square tiled by matching τ-tiles. Every side of a macro-tile carries a sequence of N colors called a *macro-color*.

Let ρ be a set of τ-macro-tiles. We say that τ *simulates* ρ if (a) τ-tilings exist, and (b) for every τ-tiling there exists a unique grid of vertical and horizontal lines that cuts this tiling into $N \times N$ macro-tiles from ρ.

Example 1. Assume that we have only one ('white') color and τ consists of a single tile with 4 white sides. Fix some N. There exists a single macro-tile of size $N \times N$. Let ρ be a singleton that contains this macro-tile. Then every τ-tiling can be cut into macro-tiles from ρ. However, τ does not simulate ρ since the placement of cutting lines is not unique.

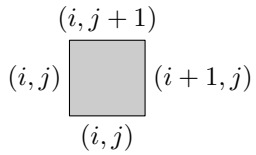

Fig. 1. Tiles of the set τ for Example 2

Example 2. In this example a set ρ that consists of a single macro-tile (that matches itself horizontally and vertically) is simulated by some tile set τ. The tile set τ consists of N^2 tiles indexed by pairs (i, j) of integers modulo N. A tile from τ has colors on its sides as shown (Fig. 1). The macro-tile in ρ has colors $(0, 0), \ldots, (0, N-1)$ and $(0, 0), \ldots, (N-1, 0)$ on its borders. (Fig. 2).

If a tile set τ simulates some set ρ of τ-macro-tiles with zoom factor $N > 1$ and ρ is isomorphic to τ, the set τ is called *self-similar* (an *isomorphism* between τ and ρ is a bijection that respects the relations "one tile can be placed on the right of another one" and "one tile can be placed on the top of another one").

The idea of self-similarity is used (more or less explicitly) in most constructions of aperiodic tile sets ([8, 2] are exceptions). The usage of self-similarity is based on the following remark:

Proposition 1 (folklore). *All self-similar tile sets τ have only aperiodic tilings.*

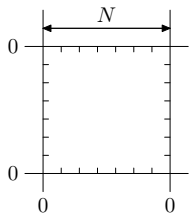

Fig. 2. Macro-tile of size $N \times N$ for Example 2

(A simple proof of this statement can be found, e.g., in [4] or [5].)

So to prove the existence of aperiodic tile sets it is enough to construct a self-similar tile set.

Theorem 3. *There exists a self-similar tile set τ.*

In the rest of this section we explain some technique (similar to the classical proof of Kleene's fixed-point theorem) that can be used to construct self-similar tile sets. In particular, we get a proof of Theorem 3. In the sequel we generalize this argument and use it in more complicated situations.

First of all, we explain some technique used in our construction: how to simulate a given tile set by embedding computations.

2.2 Simulating a tile set

For brevity we say that a tile set τ simulates a tile set ρ when τ simulates some set of macro-tiles $\tilde{\rho}$ isomorphic to ρ (e.g., we say that a self-similar tile set simulates itself).

Let us start with some informal discussion. Assume that we have a tile set ρ whose colors are k-bit strings ($C = \{0,1\}^k$) and the set of tiles $\rho \subset C^4$ is presented as a predicate $R(c_1, c_2, c_3, c_4)$ of four k-bit arguments. Assume that we have some Turing machine $\mathcal{R}$ that computes R. Let us show how to simulate ρ using some other tile set τ.

This construction extends Example 2, but simulates a tile set ρ that contains not a single tile but many tiles. We keep the coordinate system modulo N embedded into tiles of τ; these coordinates guarantee that all τ-tilings can be uniquely cut into blocks of size $N \times N$ and every tile "knows" its position in the block (as in Example 2). In addition to the coordinate system, now each tile in τ carries supplementary colors (from a finite set specified below) on its sides. These colors form a new "layer" superimposed with the old one, i.e., the set of colors is now a Cartesian product of the old one and the set of colors used in this layer.

On the border of a macro-tile (i.e., when one of the coordinates is zero) only two supplementary colors (say, 0 and 1) are allowed. So the macro-color encodes a string of N bits (where N is the size of macro-tiles). We assume that $N \geq k$ and let k bits in the middle of macro-tile sides represent colors from C. All other

bits on the sides are zeros (this is a restriction on tiles: each tile "knows" its coordinates so it also knows whether non-zero supplementary colors are allowed).

Now we need additional restrictions on tiles in τ that guarantee that macro-colors on the sides of each macro-tile satisfy the relation R. To achieve this, we ensure that bits from the macro-tile sides are transferred to the central part of the tile where the checking computation of $\mathcal{R}$ is simulated.

For that we need to fix which tiles in a macro-tile form "wires" (this can be done in any reasonable way; let us assume that wires do not cross each other) and then require that each of these tiles carries equal bits on two sides (so some bit propagates along the entire wire); again it is easy to arrange since each tile knows its coordinates.

Then we check R by a local rule that guarantees that the central part of a macro-tile represents a time-space diagram of $\mathcal{R}$'s computation (the tape is horizontal, time goes up). This is done in a standard way. We require that computation terminates in an accepting state: if not, the tiling cannot be formed.

To make this construction work, the size of macro-tile (N) should be large enough: we need enough space for k bits to propagate and enough time and space (=height and width) for all accepting computations of $\mathcal{R}$ to terminate.

In this construction the number of supplementary colors depends on the machine $\mathcal{R}$ (the more states it has, the more colors are needed in the computation zone). To avoid this dependency, we replace $\mathcal{R}$ by a fixed universal Turing machine $\mathcal{U}$ that runs a *program* simulating $\mathcal{R}$. Let us agree that the tape of the universal Turing machine has an additional read-only layer. Each cell carries a bit that is not changed during the computation; these bits are used as a program

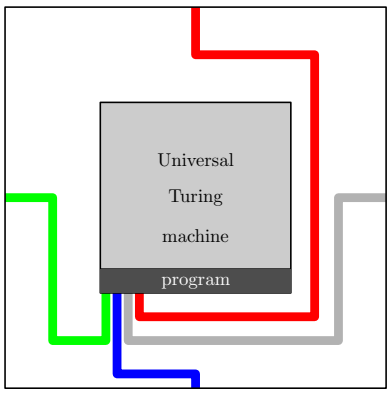

Fig. 3. Checking tiles with a universal TM

for the universal machine $\mathcal{U}$ (Fig. 3). In terms of our simulation, the columns of the computation zone carry unchanged bits (considered as a program for U), and the tile set restrictions guarantee that the central zone represents the protocol

of an accepting computation of $\mathcal{U}$ (with this program). In this way we get a tile set τ that simulates ρ with zoom factor N using $O(N^2)$ tiles. (Again we need N to be large enough, but the constant in $O(N^2)$ does not depend on N.)

2.3 Simulating itself

We know how to simulate a given tile set ρ (represented as a program for the universal TM) by another tile set τ with a large enough zoom factor N. Now we want τ to be isomorphic to ρ (then Proposition 1 guarantees aperiodicity). For this we use a construction that follows Kleene's recursion (fixed-point) theorem [9].

Note that most rules of τ do not depend on the program for $\mathcal{R}$, dealing with information transfer along the wires, the vertical propagation of unchanged program bits, and the space-time diagram for the universal TM in the computation zone. Making these rules a part of ρ's definition (we let $k = 2\log N + O(1)$ and encode $O(N^2)$ colors by $2\log N + O(1)$ bits), we get a program that checks that macro-tiles behave like τ-tiles in this respect.

The only remaining part of the rules for τ is the hardwired program. We need to ensure that macro-tiles carry the same program as τ-tiles do. For that our program (for the universal TM) needs to access the bits of its own text. (This self-referential action is in fact quite legal: the program is written on the tape, and the machine can read it.) The program checks that if a macro-tile belongs to the first line of the computation zone, this macro-tile carries the correct bit of the program.

How should we choose N (hardwired in the program)? We need it to be large enough so the computation described (which deals with $O(\log N)$ bits) can fit in the computation zone. The computation is rather simple (polynomial in the input size, i.e., $O(\log N)$), so for large N it easily fits in $\Omega(N)$ available time.

This finishes the construction of a self-similar aperiodic tile set.

2.4 Variable zoom factor

This construction is flexible enough and can be used in other contexts. For example, the "zoom factor" N could depend on the level k (i.e., be different for macro-tiles, macro-macro-tiles etc.) For this each macro-tile should have k encoded at its sides; this labeling should be consistent when switching to the next level. Using the anthropomorphic terminology, we say that each macro-tile "knows" its level, i.e., the sequence of bits that form a binary representation of the level is transferred from the sides to the tape and the computation checks that all these numbers (level bits for all four sides) are the same. This is, so to say, a "conscious" information processed by a computation in the central region of the macro-tile. One may say also that a macro-tile of any level contains "subconscious" information ("existing in mind but not immediately available to consciousness", as the dictionary says): this is the information that is conscious for the sub-tiles that form a macro-tile, and their sub-tiles (all the way down to the ground level).

Using this terminology, we can say that each macro-tile knows its coordinates in the macro-tile of the next level: for a tile of level k these coordinates are integers modulo N_{k+1}, so in total $\log k + O(\log N_{k+1})$ bits are required for keeping both the level and these coordinates. Note that N_k steps should be enough to perform increment operation modulo N_{k+1}; we assume that both $\log k$ and $\log N_{k+1}$ are much less than N_k. This means that N_k should not increase too fast or too slow (say, $N_k = \log k$ is too slow and $N_{k+1} = 2^{N_k}$ is too fast). Also we need to compute N_{k+1} when k is known, so we assume that not only the size of N_{k+1} (i.e., $\log N_{k+1}$) but also the time needed to compute it given k are small compared to N_k. These restrictions still allow many possibilities, say, $N_k = \sqrt{k}$, $N_k = k$, $N_k = 2^k$, $N_k = 2^{(2^k)}$, $N_k = k!$ etc.

There is one more important point that needs to be covered. How do we guarantee that the bits representing the level k (on the tape of a macro-tile) are correct? In other terms, we need to ensure that the levels known to a macro-tile and to one of its tiles differ by one. (In psychoanalytic terms we need to check that conscious and subconscious information in a tile match each other.) This is done as follows. The tile knows its level and also knows its position in the macro-tile it belongs (its father). So it knows whether it is in the place where father should keep level bits, and can check whether indeed the level bit that father keeps in this place is consistent with the level information the tile has.

3 Tile set that has only complex tilings

In this section we provide a new proof of Theorem 2.

3.1 A bi-infinite bit sequence

Proof. We start the proof in the same way as in [3]: we assume that each tile keeps a bit that propagates (unchanged) in the vertical direction. Then any tiling contains a bi-infinite sequence of bits ω_i (where $i \in \mathbb{Z}$). Any $N \times N$ square contains a N-bit substring of this string, so if (for large enough N) any N-bit substring of ω has complexity at least $c_1 N$ for some fixed c_1, we are done.

Such a bi-infinite sequence indeed exists (see [3]; another proof can be obtained by using Lovasz local lemma, see [15]). So our goal is to formulate tilings rules in such a way that a correct tiling "ensures" that the bi-infinite sequence embedded in it indeed has this property.

The set of all "forbidden" binary strings, i.e., strings x such that $K(x) < c_1|x| - c_2$ (here $K(x)$ stands for Kolmogorov complexity of x and $|x|$ stands for the length of x) is enumerable: there is a program that generates all forbidden strings. It would be nice to embed into the tiling a computation that runs this program and compares its output strings with the substrings of ω; such a computation may blow up (create a tiling error) if a forbidden substring is found.

However, this is not easy. There are several difficulties.

- First of all, our self-similar tiling contains only finite computations; the duration depends on the zoom factor and may increase as the level increases

(bigger macro-tiles keep longer computations), but at any level the computations are finite.
– The computation at some level deals with bits encoded in the cells of that level, i.e., with macro-tile states. So the computation cannot access the bits of the sequence directly (they are "deep in the subconscious"), and some mechanism to dig them out is needed.

Let us explain how to overcome these difficulties.

3.2 Bits delegation

Macro-tile of level k is a square whose side is $L_k = N_0 \cdot N_1 \cdot \ldots \cdot N_{k-1}$, so there are L_k vertical lines (carrying the bits of the sequence) that intersect this macro-tile. Let us delegate each of these bits to one of the macro-tiles of level k that cover the corresponding line. Note that a macro-tile of the next $(k+1)$-th level is made of $N_k \times N_k$ macro-tiles of level k. We assume that N_k is much bigger than L_k (more about choice of N_k later); this guarantees that there is enough macro-tiles of level k (in the next level macro-tile) to serve all bits that intersect them. Let us decide that ith macro-tile of level k (from bottom to top) in a $(k+1)$-level macro-tile serves (consciously knows, so to say) (i mod L_k)-th bit (from the left) in its zone. (In this way we have several macro-tiles of level k in each macro-tile of level $k+1$ that are responsible for the same bit, but this does not create any problems.)

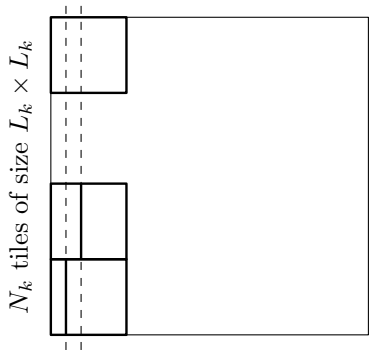

Fig. 4. Bit delegation

So each bit (each vertical line) has a representative on every level — a macro-tile that consciously knows this bit. However, we need some mechanisms that guarantee that this information is indeed true (consistent on different levels). On the bottom level it is easy, since the bits are available directly.

To guarantee the consistency we use the same trick as in Section 2.4: at each level we keep the information not only for this level but also for the next

(father) level, and made necessary consistency checks. Namely, each macro-tile knows (has on its computation tape):

- the bit delegated to this macro-tile;
- the coordinates of this macro-tile in its father macro-tile (that are already used in the fixed-point construction); the y-coordinate determines the position of the bit delegated to this macro-tile (relative to the left boundary of the macro-tile).
- the bit delegated to the father of this macro-tile;
- the coordinates of the father macro-tile in the grandfather macro-tile.

This information is subject to consistency checks:

- the information about the father macro-tile should coincide with the same information in neighbor tiles (unless they have a different father, i.e., one of the coordinates is zero).
- if it happens that the bit delegated to the father macro-tile is the same bit as delegated for the tile, these bits should match;
- it can happen that the macro-tile occupies a place in its father macro-tile where some bits of its coordinates (inside grandfather macro-tile) or the bit delegated to the father are kept; then this partial information on the father level should be checked against the information about father coordinates and bit.

These tests guarantee that the information about father is the same in all brothers, and some of these brothers (that are located on the father tape) can check it against actual father information; at the same time some other brother (that has the same delegated bit as the father) checks the consistency of the delegated bits information.

Note that this scheme requires that both $\log N_k$ and $\log N_{k+1}$ are much smaller than N_{k-1}. This is the case, for example, if $N_k = 2^{4^k}$; note that $L_k = N_0 \cdot N_1 \cdot \ldots \cdot N_{k-1}$ is then less than $\sqrt{N_k}$ (which is even better than the requirement $L_k \leq N_k$ mentioned earlier).

In Section 4 we set $N_k = Q^{2.5^k}$ for some large enough Q. (In fact, any constant between 2 and 3 can be used instead of 2.5.)

3.3 Bit blocks checked

We explained how macro-tile of any level can get a true information about one bit (delegated to it). However, we need to check not bits, but substrings (and create a tiling error if a forbidden string appears). Note that it is OK to test only very short substrings compared to the macro-tile size (N_k): if this test is done on all levels, this short substring becomes long enough to detect any violation. (Also note the short forbidden substrings can appear very late in the generation process, so we need computation in arbitrary high levels for this reason, too.)

So we need to provide more information to tiles. It can be done in the following way. Let us assume that a tile contains not one bit but a group of bits

that starts at the delegated bit and has length depending on the level k (and growing very slowly with k, say, $\log\log\log k$ is slow enough). If this group goes out of the region occupied by a tile, we truncate it.

Similarly, a macro-tile should have this information for the father macro-tile (even if the bits are outside its own region), this information should be the same for brothers and needs to be checked against the delegated bits on the macro-tile level and pieces of information on the father level.

Then the computation in the computation zone can start the generating process checking the forbidden strings that appear against all the substrings of the group of bits available to this computation. This process is time- and space-bounded, but this does not matter since every string if considered on a high enough level.

3.4 Last correction

The argument explained above still needs some correction. We claim that every forbidden string will be detected at some level where it is short enough compared to the level parameters. However, there could be strings that never become a part of one macro-tile. Imagine that there is some vertical line that is a boundary between macro-tiles of all levels (so we have bigger and bigger tiles on both sides, and this line still separates them). Then a substring that crosses this line will be never checked and therefore we cannot guarantee that it is not forbidden.

There are several ways to get around this problem. One can decide that each macro-tile contains information not only about blocks inside its father macro-tile but in a wider regions (say, three times wider including uncle macro-tiles); this information should be checked for consistency between cousins, too.

But there is a simpler solution. Note that even if a string that crosses the boundary is never checked, its parts (on both sides of the boundary) are, so their complexity is proportional to their length. And one of the parts has length at least half of the original length, so we still have a complexity bound, just the constant is twice smaller.

This finishes the proof of Theorem 2. □

4 Robust tile set that enforces complex tilings

We want to construct a "robustified" tile set such that any tiling with "sparse enough" errors or holes can be patched (by changing a small fraction of tiles). It does not matter whether we speak about errors (places where two neighbor tiles do not match) or holes (places without tiles). Indeed, we can convert a tiling error into a hole by deleting one of the two non-matching tiles and convert a hole into a small number of errors by placing an arbitrary tile there.

Let E be a subset of $\mathbb{Z}^2$ and let τ be a tile set.

Definition 1. *A (τ, E)-tiling is a mapping*

$$T\colon\ (\mathbb{Z}^2 \setminus E) \to \tau$$

such that for every two neighbor cells $x, y \in \mathbb{Z}^2 \setminus E$, the tiles $T(x)$ and $T(y)$ match.

In other terms, T is a τ-tiling of the complement of E.

We assume that a set of errors E is chosen at random, according to the Bernoulli distribution B_ε: every point in $\mathbb{Z}^2$ belongs to E with some probability $\varepsilon > 0$; the random choices at different points are done independently.

Theorem 4. *There exist a tile set τ and constants $c_1, c_2 > 0$ with the following properties:*

(1) *a τ-tiling of $\mathbb{Z}^2$ exists;*

(2) *for every τ-tiling T of the plane, every $N \times N$-square in T has Kolmogorov complexity at least $c_1 N - C_2$;*

(3) *for all sufficiently small ε for almost every (with respect to the Bernoulli distribution B_ε) subset $E \subset \mathbb{Z}^2$ every (τ, E)-tiling is at most $1/10$ Besicovitch-apart from some τ-tiling of the entire plane $\mathbb{Z}^2$;*

(4) *for all sufficiently small ε for almost every (with respect to B_ε) subset $E \subset \mathbb{Z}^2$ and every (τ, E)-tiling T, Kolmogorov complexity of centered frames of T of size $n \times n$ is $\Omega(n)$.*

Note that in (4) we speak about complexity of squares with "holes" understood as the minimal complexity for all possible ways to fill the holes. Note also that we cannot claim that every $n \times n$ square has high complexity since this square can be completely isolated from the rest of the tiling by elements of E and therefore can be simple: lexicographically first tiling of the $n \times n$ square has complexity $O(\log n)$.

This is the main result of our paper. Its proof is based on a generalization of the construction from Section 3. Here we outline the plan of the proof:

- **bi-islands (probabilistic part):** we prove that with probability 1 random errors can be split into isolated 'doubled islands' of different rank (an n-level doubled island, or a *bi-island*, consists of two sets of diameter $O(Q^{2.5^n})$ and is isolated from other bi-islands of the same rank). This construction slightly improves the technique used in [5] (the general idea of splitting a random sparse set in ranked 'islands' goes back to [7]).
- **robustification (combinatorial part):** we embed into the primary structure of a self-similar tiling (with variable zoom factors) some redundancy, so that every single isolated island (and even a bi-island) of errors can be patched. The patching procedure involves correction of the tiling only in a small neighborhood of the island (bi-island). So we can sequentially 'patch' all errors, starting from bi-islands of low rank.
- **checksums (error-correction trick):** high level macro-tiles calculate some checksums for the bits in their 'subconscious' and communicate them to their neighbors. This guarantees that most macro-tiles on all levels have coherent bits in their subconscious, even if there are sparse errors.
- **patching errors (join everything together):** we check that with probability 1, for a randomly chosen set of errors E every tiling of $\mathbb{Z}^2 \setminus E$ can

be converted into a (close enough) tiling of the entire plane. Composing this fact with the proof from Section 3 we get the theorem.

The full proof of this theorem is rather technical, and exceeds the conference paper limits.

References

1. R. Berger, The Undecidability of the Domino Problem. *Mem. Amer. Math. Soc.*, **66**, 1966.
2. K. Culik, An Aperiodic Set of 13 Wang Tiles, *Discrete Math.*, **160**, 245–251, 1996.
3. B. Durand, L. Levin, A. Shen, Complex Tilings. *J. Symbolic Logic*, **73** (2), 593–613, 2008 (See also Proc. *33rd Ann. ACM Symp. Theory Computing*, 732–739, 2001.) Online version: `www.arxiv.org/cs.CC/0107008`
4. B. Durand, L. Levin, A. Shen, Local Rules and Global Order, or Aperiodic Tilings, *Math. Intelligencer*, **27**(1), 64–68, 2004.
5. B. Durand, A. Romashchenko, A. Shen, Fixed Point and Aperiodic Tilings, *Developments in Language Theory, 12th International Conference, DLT 2008, Kyoto, Japan, September 16–19, 2008. Proceedings*, Springer, Lecture Notes in Computer Science, volume 5257, 2008. Online version: `http://arxiv.org/abs/0802.2432`
6. P. Gács, Reliable Cellular Automata with Self-Organization, In Proc. *38th Ann. Symp. Found. Comput. Sci.*, 90–97, 1997.
7. P. Gács, Reliable Cellular Automata with Self-Organization, *J. Stat. Phys.*, **103** (1/2), 45–267, 2001.
8. J. Kari, A Small Aperiodic Set of Wang tiles, *Discrete Math.*, **160**, 259–264, 1996.
9. H. Rogers, *The Theory of Recursive Functions and Effective Computability*, Cambridge, MIT Press, 1987.
10. G. Lafitte, M. Weiss: Computability of Tilings. IFIP TCS 2008, 187-201.
11. L. Levin, Aperiodic Tilings: Breaking Translational Symmetry, *Computer J.*, **48**(6), 642–645, 2005. On-line: `http://www.arxiv.org/cs.DM/0409024`
12. J. von Neumann, *Theory of Self-reproducing Automata*, Edited by A. Burks, University of Illinois Press, 1966.
13. R. Robinson, Undecidability and Nonperiodicity for Tilings of the Plane. *Inventiones Mathematicae*, **12**, 177–209, 1971.
14. L. Bienvenu, A. Romashchenko, A. Shen. Sparse sets. *Journées Automates Cellulaires 2008* (*Uzès*), p. 18–28. MCCME Publishers, 2008. Available online: http://hal.archives-ouvertes.fr/docs/00/27/40/10/PDF/18-28.pdf
15. An. Rumyantsev, M. Ushakov, Forbidden Substrings, Kolmogorov Complexity and Almost Periodic Sequences, *STACS 2006 Proceedings*, Lecture Notes in Computer Science, Vol. 3884, Springer, 2006.