Polyominoes Simulating Arbitrary-Neighborhood Zippers and Tilings

Lila Kari, Benoît Masson

Polyominoes Simulating Arbitrary-Neighborhood Zippers and Tilings[☆,☆☆]

Lila Kari[a,*], Benoît Masson[a]

[a] *University of Western Ontario – Department of Computer Science*
Middlesex College, London, Ontario, Canada, N6A 5B7

Abstract

This paper provides a bridge between the classical tiling theory and cellular automata on one side, and the complex neighborhood self-assembling situations that exist in practice, on the other side.

A neighborhood N is a finite set of pairs $(i, j) \in \mathbb{Z}^2$, indicating that the neighbors of a position (x, y) are the positions $(x + i, y + j)$ for $(i, j) \in N$. This includes classical neighborhoods of size four, as well as arbitrarily complex neighborhoods. A generalized tile system consists of a set of tiles, a neighborhood, and a relation which dictates which are the "admissible" neighboring tiles of a given tile. Thus, in correctly formed assemblies, tiles are assigned positions of the plane in accordance to this relation.

We prove that any path filled with tiles defined in a given but arbitrary neighborhood (a zipper) can be simulated by a simple "ribbon" of microtiles. A ribbon is a special kind of polyomino, consisting of a non-self-crossing rectilinear sequence of tiles on the plane, in which successive tiles are adjacent along an edge, and where each tile needs to match glues with only two other tiles: its predecessor and its successor on the path. Our constructions simulate each of the existing tiles by a polyomino of microtiles, whose shape is used to simulate the given tile and the communication of information between itself and its neighbors. The polyominoes can then be catenated together to simulate the entire complex-neighborhood tiled path by a continuous two-tile-neighborhood ribbon.

Finally, we extend this construction to the case of traditional tilings, proving that we can simulate arbitrary-neighborhood tilings by simple-neighborhood tilings, while preserving some of their essential properties.

Keywords: DNA computing, self-assembly, tilings, polyominoes

[☆] Some results in Section 3 of this paper were presented at FNANO 2009 conference [16].

[☆☆] This research was supported by a Natural Sciences and Engineering Research Council of Canada discovery grant, and by a Canada Research Chair award to L.K.

[*] Corresponding author.
Email addresses: lila@csd.uwo.ca (Lila Kari), benoit@csd.uwo.ca (Benoît Masson)

1. Introduction

Because of the constant miniaturization of components, microscopic elements in the fields of electronics, engineering or even medicine are becoming more and more difficult to construct and to manipulate. A recent approach to work around this problem is *self-assembly*. It consists in "programming" the nano-components, so that when starting from an initial pool of selected components, they automatically aggregate to form bigger and bigger structures, until they eventually form a final complex structure.

The first formal models for self-assembly were introduced a decade ago [24, 26, 1]. In this framework, self-assembling components were modelled by Wang tiles [23], i.e., unit squares that cannot be rotated and that have "glues" on each of their four edges. Two tiles stick to each other if they have the same glue on their common edge. By carefully designing the glues, and starting with an initial tile called "seed", complex structures can self-assemble.

The use of this simple model as a formalization of the process of self-assembly allowed the application for theoretical studies of dynamical self-assembly of many well-known existing results and techniques concerning static tilings [18] and cellular automata [15], such as the undecidable problem of the tiling of the plane [8], and the simulation of a Turing machine by a tile system [18].

Most of the theoretical results on self-assembly presume that each tile interacts via glues only with tiles in its so-called *von Neumann neighborhood*, which includes the four tiles situated at the North, South, East, and West of the tile itself [21]. Only relatively few recent results consider more general cases, such as larger neighborhood [4], or a three-dimensional neighborhood [7]. Even the most well-known experimental incarnation of square tiles, the DNA tiles [26, 27, 17, 20, 13], deal only with the von Neumann-sized neighborhood, where the DNA single strands located at the corner of each rectangular DNA tile allow its interaction with four neighbors only. Other experimental situations that could be modelled by self-assembly of tiles, such as atomic or molecular interactions, potentially include more complex scenarios where the neighborhood of a tile is both larger and more complex than the von Neumann neighborhood. At the limit, one can consider the case of an arbitrary neighborhood where tiles that are not physically adjacent to the main tile may be its neighbors, while some adjacent tiles may not.

In [3, 4], it was proved that, for any directed tile system, any von Neumann-neighborhood "zipper" (a tiled rectilinear path) can be simulated by a "ribbon" constructed with tiles from a new tile system. A ribbon is a non-self-crossing rectilinear succession of tiles in the plane, where each tile is required to have glues matching with two tiles only: its predecessor and its successor on the ribbon-path. The construction that simulated a directed von Neumann-neighborhood tiled path by a ribbon, replaced each of the existing tiles by so-called "motifs" which traced the contours of the initial tile but where, in addition, bumps and matching dents along the motif edges simulated both the matching of the glues and the directionality of the path. In other words, geometry of the motifs was used to simulate glue matching. Note that motifs, as well as ribbons, are

particular cases of *polyominoes* [22, 6], i.e., finite and connected sets of DNA tiles.

This polyomino construction led to a conjecture by Jarkko Kari, claiming that it is possible to "simulate" an arbitrary-neighborhood zipper by a simple two-tile-neighborhood ribbon. A first step in this direction was [10, 11], wherein it was proved that a complex-neighborhood zipper, defined for example on the Moore neighborhood (von Neumann plus four diagonal neighbors), can be simulated by a ribbon of irregularly shaped tiles, where the shape was used to simulate the neighborhood relationship.

The aim of this paper is to answer the above conjecture positively for the case of arbitrary-neighborhood zippers, thus providing a bridge between the classical work in tiling theory or cellular automata and the realistic complex-neighborhood self-assemblies that exist in practice. We namely prove in Corollary 3.10 that for *any* neighborhood, zippers can be simulated by simple polyominoes connected to each other end-to-end to form a ribbon that essentially traces the same path. The main idea used in our simulation is that each existing tile can be replaced by a polyomino, where the shape of the polyomino is used to simulate the communication between a tile and its adjacent or distant neighbors. We also show that, by the design of the shapes of the polyominoes, we can transmit information at a distance, sometimes across other information pathways, without violating the non-self-crossing feature of the ribbon. Such situations where information pathways cross are inherent in, for example, Moore neighborhoods where, e.g., the communication channel between a tile and its Northeast neighbor "crosses" the communication channel between its North neighbor and its East neighbor.

We also explain how the simulation of zippers by ribbons can be modified to simulate arbitrary-neighborhood tilings by von Neumann-neighborhood tilings. The idea is to modify the polyominoes, adding new constraints so that the two-tile neighborhood can be replaced by a von Neumann neighborhood (Corollary 4.3). The main significance of this simulation, as opposed to more intuitive ones where some "supertiles" are used to transfer information, is that it applies to all tilings, even when they are partial. Besides, some essential properties of the initial tiling are preserved by the simulation. We prove that this is the case for partial, periodic, line or column convex tilings.

The paper is organized as follows. In Sect. 2, we recall basic definitions and give a formal definition of a simulation. Then, Sect. 3 describes our construction in the case of zippers, starting with the general idea of simulating a zipper by a ribbon, by sketching the proof from [3, 4], in the simple case of a von Neumann neighborhood. We also highlight the technical difficulties related to crossing of information pathways, that prevent this technique from being transferable without modifications to the case of the Moore neighborhood [10]. Then, we prove our result for the case of arbitrary linear neighborhoods. This construction is eventually generalized to arbitrary neighborhoods to prove the main result of this section. Finally, in Sect. 4, we adapt the construction to the simulation of regular tilings, and we prove that partial, periodic, line or column convex

arbitrary-neighborhood tilings can be simulated by equivalent von Neumann-neighborhood tilings.

2. Definitions

First, we give some basic definitions on tilings, and then we introduce more technical definitions to formalize the notion of simulation by polyominoes.

2.1. Tilings, Ribbons, and Zippers

Historically, Wang tiles [23] were defined as oriented unit squares, on the border of which were 4 *glues* (colors) used to stick them to neighboring tiles, provided the glues matched. We generalize this notion to arbitrary neighborhoods [15, 4], i.e., neighborhoods which can contain other tiles than the North, South, West and East tiles. In this paper, a *neighborhood* $N \subset \mathbb{Z}^2$ is the set of relative coordinates of the neighboring tiles, such that

- $|N| < \infty$ (finite number of neighbors);
- $(0,0) \notin N$ (a tile can not be one of its neighbors);
- $(i,j) \in N \Rightarrow (-i,-j) \in N$ (if a tile t' is a neighbor of a tile t, then t has to be a neighbor of t').

For example, the usual 4-tile neighborhood (called *von Neumann* neighborhood [15]) is $N_{vN} = \{(0,1),(1,0),(0,-1),(-1,0)\}$; the 8-tile *Moore* neighborhood is $N_M = \{(0,1),(1,1),(1,0),(1,-1),(0,-1),(-1,-1),(-1,0),(-1,1)\}$.

For the following definitions, a neighborhood N and a finite set of glues X are fixed. A *tile* is a tuple $t = (t_{i,j})_{(i,j)\in N}$ where each $t_{i,j} \in X$. Intuitively, $t_{i,j}$ is the glue that will be used to match the neighbor located at position (i,j) relative to the tile t. A *tile system* T is a finite set of tiles, used to build larger structures. A tile $t \in T$ *sticks* at position (i,j) to a tile $t' \in T$ if the corresponding glues match, i.e., $t_{i,j} = t'_{-i,-j}$.

A (T,N)-*tiling* using tile system T in neighborhood N (or N-*neighborhood tiling*, or simply *tiling* when the tile system and its neighborhood are known without ambiguity) is a mapping $\tau : D \to T$, where $D \subset \mathbb{Z}^2$ is a subset of the plane, which associates every position $(x,y) \in D$ with a tile, such that all tiles stick to their neighbors, i.e., for all $(x,y) \in D$, for all $(i,j) \in N$ such that $(x+i, y+j) \in D$, $\tau(x,y)_{i,j} = \tau(x+i, y+j)_{-i,-j}$. Note that tilings are often referred to as "valid" tilings in the literature. The set of all (T,N)-tilings is denoted $\mathfrak{T}_{T,N}$.

When $D = \mathbb{Z}^2$, the tiling is said to be *total*, otherwise it is *partial*. In addition, if $|D| < \infty$ then the partial tiling is also *finite*. A tiling $\tau : D \to T$ is *connected* if its domain D is connected. As suggested in [22], we call *polyomino* a finite and connected tiling. Note that the usual definition of a polyomino (see for example [6]) refers to a domain $D \subset \mathbb{Z}^2$, while here we add to this notion a mapping to tiles.

According to the usual definition, a total tiling τ is *periodic* if it admits a horizontal and a vertical *period* $p_h, p_v \in \mathbb{N}_+$, i.e., for all $x, y \in \mathbb{Z}$, $\tau(x, y) = \tau(x+p_h, y) = \tau(x, y+p_v)$. We extend this definition to partial tilings as follows: a tiling $\tau : D \to T$ is periodic if it admits a horizontal and a vertical period $p_h, p_v \in \mathbb{Z}$ such that for all $(x, y) \in D$ and for all $\alpha, \beta \in \mathbb{Z}$, $(x+\alpha p_h, y+\beta p_v) \in D$ implies $\tau(x, y) = \tau(x+\alpha p_h, y+\beta p_v)$. Note that with this extended definition, all finite tilings are periodic (the periods should be "larger" than the tiling itself).

A tiling $\tau : D \to T$ is *line convex* [resp., *column convex*] if $(x_1, y), (x_2, y) \in D$ and $x_1 < x_2$ [resp., $(x, y_1), (x, y_2) \in D$ and $y_1 < y_2$] imply that for all $x_1 < x < x_2$ [resp., for all $y_1 < y < y_2$], $(x, y) \in D$. A tiling is *convex* if it is both line and column convex.

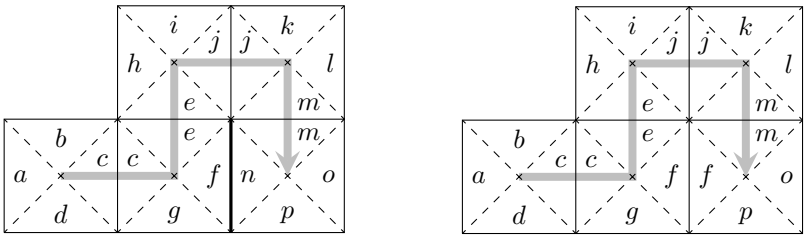
Two positions of the plane $u, v \in \mathbb{Z}^2$ (or, by extension, tiles of a tiling) are said to be *adjacent* when they are neighbors in the von Neumann sense, i.e., $v - u \in N_{vN}$. A *path* is a sequence of adjacent positions of the plane. Formally, a path is a function $P : I \to \mathbb{Z}^2$, where $I \subset \mathbb{Z}$ is a set of consecutive integers, such that for all $i, i+1 \in I$, $P(i)$ and $P(i+1)$ are adjacent. For a given tile system T, a *T-tiled path* using tile system T is a sequence of adjacent tiles from T, i.e., a pair (P, r) where P is a path and $r : \mathrm{range}(P) \to T$ a mapping from positions to tiles. We say that a T-tiled path is *finite* when $|\mathrm{dom}(P)| < \infty$, and we may also call a finite T-tiled path a *polyomino* since it is connected (by definition of the path).

A T-tiled path (P, r) is a *T-ribbon* using tile system T (simply called *ribbon* when T is known without ambiguity) if P is injective (non-self-crossing) and for all $i, i+1 \in \mathrm{dom}(P)$, $r(P(i))_v = r(P(i+1))_{-v}$, with $v = P(i+1) - P(i)$. The glue $r(P(i))_v$ is called the *output* glue of $r(P(i))$, while $r(P(i+1))_{-v}$ is the *input* glue of $r(P(i+1))$. Informally, a ribbon is a sequence of adjacent tiles which stick to their predecessor and successor only (see Fig. 1(a)). Consequently, r is not necessarily a tiling.

A T-tiled path (P, r) is a *(T, N)-zipper* using tile system T in neighborhood N (or *N-neighborhood zipper*, or *zipper* when T and N are known) if P is injective and r is a (T, N)-tiling. A zipper can be seen as a tiling with an additional notion of unique input and output for every tile (except the first which has only an output and the last which has only an input), each input being connected to the output of an adjacent tile (see Fig. 1(b)). Note that zippers can be defined in arbitrary neighborhoods, since it is not required that adjacent glues match. The set of T-ribbons is denoted $\mathfrak{R}_T$, the set of (T, N)-zippers $\mathfrak{Z}_{T,N}$.

2.2. Poly-Tilings and Simulations

In this section, we give a formal definition of the intuitive notion of the simulation of a tiling by another, as well as the simulation of a zipper by a ribbon. Informally, a simulation is an injective function which associates each tiling [resp., zipper] using the "initial" tile system in the "initial" neighborhood with a tiling [resp., ribbon] using a "new" tile system in a "new" neighborhood. The injectivity of the simulation allows to recover the initial object without



(a) A ribbon. (b) A zipper.

Figure 1: A ribbon, and a von Neumann-neighborhood zipper. The underlying path is drawn in gray. The only difference occurs at the bottom-right where glues f and n are not required to match in the ribbon, while in the zipper they do.

ambiguity. To build this new object, we first associate every initial tile with unique polyominoes. Then, we catenate these polyominoes to form the new, bigger, object, called *poly-tiling* or *poly-ribbon*.

Definition 2.1. A *poly-tiling* τ of *scale* s, using tile system T in neighborhood N, is a mapping $\tau : D \to \mathfrak{T}_{T,N}$, with $D \subset \mathbb{Z}^2$, such that

(*i*) for all $u \in D$, $\tau(u)$ is a finite (T, N)-tiling;

(*ii*) for all $u, v \in D$, $u \neq v$, $[\mathrm{dom}(\tau(u)) + su] \cap [\mathrm{dom}(\tau(v)) + sv] = \emptyset$;

(*iii*) for all $u, u' \in D$, $v \in \mathrm{dom}(\tau(u))$ and $v' \in \mathrm{dom}(\tau(u'))$, if $(i, j) = (su' + v') - (su + v) \in N$, then $\tau(u)(v)_{i,j} = \tau(u')(v')_{-i,-j}$.

Let us discuss the items in this definition from an informal point of view, as shown in Fig. 2. We refer only to the particular case in which all $\tau(u)$ are connected and thus polyominoes, since this is what we will construct. In this case, the definition implies that a poly-tiling of scale s is

(i) a juxtaposition of polyominoes on a grid of size s;

(ii) such that all these polyominoes do not overlap once shifted to their actual position on the grid, which is achieved by shifting $\tau(x, y)$ by sx units to the right and sy units upwards;

(iii) neighboring polyominoes stick, i.e., if two tiles of two different polyominoes become neighbors after being shifted, they have to stick.

These characteristics allow to consider them as usual "valid" tilings without overlaps, as explained in the following remark.

Remark 2.1. A poly-tiling τ of scale s can be easily transformed into a "regular" tiling τ', as depicted in Fig. 2, according to the following formula:

$$\tau'(i, j) = (\tau(x, y))(i', j') \ ,$$

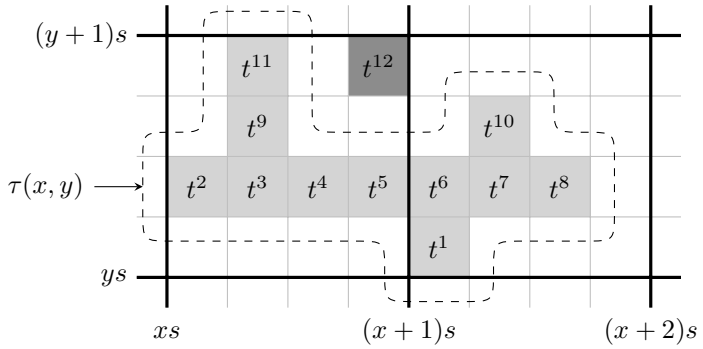


Figure 2: Poly-tiling τ of scale $s = 4$ seen as a tiling. The dashed line surrounds the tiling $\tau(x, y)$. Note that some tiles from $\tau(x, y)$ may be located outside of the "box" of size s situated at (sx, sy), and conversely that some tiles of this box (here, t^{12}) may not belong to $\tau(x, y)$.

for any $x, y, i, j, i', j' \in \mathbb{Z}$ which verify $i = sx + i'$, $j = sy + j'$ and $(i', j') \in \mathrm{dom}(\tau(x, y))$. Because of condition (ii) in the definition of poly-tilings, given any pair (i, j), if x, y, i', j' exist then they are unique. Because of (i) and (iii), τ' is indeed a tiling since all glues match.

We can slightly modify this notion to define poly-ribbons. Basically, we replace the property of sticking at the macro-level (iii) by a notion of macro-path.

Definition 2.2. A *poly-ribbon* of *scale* s, using tile system T, is a pair (P, r) where P is a path and $r : \mathrm{range}(P) \to \mathfrak{R}_T$ is a mapping such that

(*i*) for all $u \in \mathrm{range}(P)$, $r(u)$ is a finite T-ribbon;

(*ii*) for all $u, v \in \mathrm{range}(P)$, $u \neq v$, $[\mathrm{dom}(r(u)) + su] \cap [\mathrm{dom}(r(v)) + sv] = \emptyset$ (where we define $\mathrm{dom}(r(u)) = \mathrm{dom}(r')$, if $r(u) = (P', r')$);

(*iii*) for all $i, i+1 \in \mathrm{dom}(P)$, let $r(P(i)) = (P_1, r_1)$ and $r(P(i+1)) = (P_2, r_2)$ be two consecutive ribbons. Let $u_1 = P_1(\max(\mathrm{dom}(P_1)))$ be the last position of P_1, $u_2 = P_2(\min(\mathrm{dom}(P_2)))$ be the first position of P_2, and $v = [u_2 + sP(i+1)] - [u_1 + sP(i)]$ be their relative position. Then, we must have $v \in N_{vN}$ and $r_1(u_1)_v = r_2(u_2)_{-v}$.

The new requirement (iii) enforces that the ribbon follows a path at the macro-level, "jumping" from one ribbon to the other. Therefore, in a similar way as what was done in Remark 2.1, we can see a poly-ribbon as a ribbon. This explains why in our constructions we build poly-tilings [resp., poly-ribbons] for simplicity, they can in turn be considered as particular tilings [resp., ribbons] from the set $\mathfrak{T}_{T,N}$ [resp., $\mathfrak{R}_T$].

Let us now formalize the intuition that a simulation is a function ψ which associates tilings with unique poly-tilings [resp., zippers with unique poly-ribbons],

by means of an injective function φ transforming every tile into a set of polyominoes. These polyominoes will be aligned on a grid of size s, each of them replacing a tile from the initial object at the same position, to form a poly-tiling [resp., poly-ribbon] corresponding to the initial tiling [resp., zipper]. In the following, for a set S, we define $\mathcal{P}(S) = 2^S$ as the set of subsets of S.

Definition 2.3. Let $\varphi : T \to \mathcal{P}(\mathfrak{T}_{T',N'})$ be a function which associates a tile t with a set of unique finite (T', N')-tilings, such that for tiles $t \neq t'$, $\varphi(t) \cap \varphi(t') = \emptyset$. A *tiling-simulation* of the tile system T by the tile system T' is a mapping $\psi : \mathfrak{T}_{T,N} \to \mathcal{P}(\mathfrak{T}_{T',N'})$, such that ψ associates any (T, N)-tiling $\tau : D \to T$ with a subset of poly-tilings (hence tilings) from $\{\tau' : D \to \mathfrak{T}_{T',N'} \mid \tau'(x, y) \in \varphi(\tau(x, y))\}$.

Note that the initial constraint on φ implies the injectivity of φ, and therefore of ψ. Thus, given such a poly-tiling $\psi(\tau)$, one can retrieve unambiguously the initial tiling τ. Besides, in our constructions, the finite tilings $\varphi(x, y)$ will be connected *polyominoes*.

Definition 2.4. Let $\varphi : T \to \mathcal{P}(\mathfrak{R}_{T'})$ associate a tile t with a set of unique finite T'-ribbons, such that for $t \neq t'$, $\varphi(t) \cap \varphi(t') = \emptyset$. A *zipper-simulation* of the tile system T by the tile system T' is a mapping $\psi : \mathfrak{Z}_{T,N} \to \mathcal{P}(\mathfrak{R}_{T'})$, such that ψ associates any (T, N)-zipper (P, r) where $r : \text{range}(P) \to T$ with a subset of poly-ribbons (ribbons) from $\{(P, r') \mid r' : \text{range}(P) \to \mathfrak{R}_{T'}, r'(x, y) \in \varphi(r(x, y))\}$.

Provided a tiling-simulation ψ exists, we say that τ' *simulates* τ if $\tau' \in \psi(\tau)$. Similarly, for a zipper-simulation ψ, the ribbon (P', r') *simulates* the zipper (P, r) if $(P', r') \in \psi((P, r))$. Finally, when no ambiguity is possible, we will simply use the term *simulation* to refer to the simulation of tilings or zippers.

3. Simulating Arbitrary-Neighborhood Zippers by Ribbons

Here, we prove that there exists a simulation such that any zipper, using a tile system in arbitrary neighborhood, can be simulated by a ribbon using an appropriate tile system in von Neumann neighborhood. The objective of our constructions will be to define the function φ, in such a way that the polyominoes it produces are unique and do not overlap if and only if the initial tiles of the zipper stick.

The final construction being quite complex, we introduce the technical difficulties progressively. First, we recall known results which present the basic principles of the simulations, and solve the problem of crossings. Then, we deal with linear neighborhoods (all neighbors are on the same line), and finally we prove the general result in Corollary 3.10.

3.1. Preliminary Results

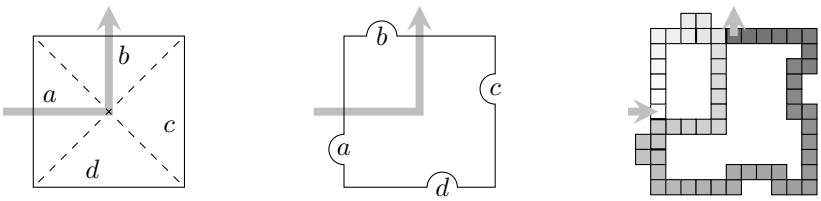
First we recall basic results of simulations of zippers. In [3, 4], the authors prove a fundamental result on the simulation of zippers by ribbons. They describe a method used to simulate bi-infinite zippers using "directed" tiles in von

Neumann neighborhood by ribbons. The result can be extended to arbitrarily long zippers and standard tiles, as recalled here.

Theorem 3.1 ([3, 4]). *Let T be a tile system in neighborhood N_{vN}, with set of glues X. There exist a simulation ψ and a tile system T_μ in neighborhood N_{vN}, with set of glues X_μ, such that any (T, N_{vN})-zipper can be simulated by a T_μ-ribbon.*

Proof. The key of the proof is the construction of the simulation φ which associates each tile with polyominoes. The basic idea behind these polyominoes is to replace every tile from T by a unique shape (Figs. 3(a) and 3(b)). Glues are replaced by *bumps* (the tile is raised) and *dents* (the tile is dug), a different glue leading to a different bump or dent. A way to uniquely code the glues is to change the vertical or horizontal position of the bump or of the dent, depending on the glue. Then, like in a jigsaw puzzle, two shapes stick if their adjacent bump and dent fit into each other. Therefore, a "ribbon" of these shapes would simulate a (T, N_{vN})-zipper, since the bumps and dents imply that the sides unconstrained by the ribbon have to match.

The second step of the construction of φ is the definition of a new tile system T_μ, in von Neumann neighborhood and with a new set of glues, which will be used to build the polyominoes simulating the initial tiles from T. This leads to a path P, starting from the middle of the side corresponding to the input direction, and leaving at the middle of the side given by the output direction (Fig. 3(c)). This path draws the contour of the shape, including bumps and dents, while the central part of the path is used to reconnect the input and output sides. Its position is determined by the point at the Southwest corner for example, which we locate at $(0, 0)$. The path is "filled" by new tiles from the set T_μ (we call these *microtiles* to avoid misunderstandings) using a mapping $r : \text{range}(P) \to T_\mu$. The finite T_μ-tiled path (P, r) is the polyomino associated with the initial tile. Remark that for one initial tile, there can be 12 different polyominoes, corresponding to the same shape but with 4 possible inputs and 3 possible outputs.



(a) Tile of the original zipper with its four glues.

(b) Coding of the glues into bumps and dents.

(c) Polyomino used to simulate the original tile.

Figure 3: Simulation of a von Neumann-neighborhood zipper using a ribbon of microtiles. The three steps of the simulation of a single tile are represented, in each one the gray arrow indicates the direction of the underlying path of the zipper.

The polyomino should be a ribbon, so we should give some details about its

construction. The first microtile (called the input microtile) has its input side colored with glue g, where g is the input glue of the initial tile; similarly the output side of the output microtile has the glue of the output side of the initial tile. For all other microtiles, the input matches the output of the previous tile, so that our polyomino (P, r) is a T_μ-ribbon. The input and output glues are unique among all the polyominoes, so this polyomino is the only possible ribbon using the tile system T_μ, once the input microtile is given. Besides, to ensure that no interference occurs, the two sides which are not colored yet have a glue that matches nothing (for example glue $null_1$ on the West or North sides, $null_2$ on the East or South sides).

For a tile $t \in T$, the set $\varphi(t)$ contains the 12 polyominoes described above. Then, obviously, if a T-tiled path is a (T, N_{vN})-zipper, one can find a "unique" poly-ribbon consisting of the catenation of polyominoes constructed by φ, which is a T_μ-ribbon. It is not really unique, since at the extremities of a finite zipper, 3 different polyominoes corresponding to the 3 possible inputs (or outputs) are admissible. Conversely, if a poly-ribbon using tile system T_μ exists, Definition 2.2 implies that

- all polyominoes do not overlap (condition (ii)), hence that the glues they simulate match on the four sides;
- the polyominoes stick on their input/output tiles (condition (iii)), hence that the polyominoes follow a path.

Using this path and these glues, one can uniquely restore the initial (T, N_{vN})-zipper. □

The following remark states an important property of our construction. In fact, the simulation ψ we just constructed can be considered as bijective.

Remark 3.1. Because of the careful design of the glues from X_μ, any T_μ-ribbon can be seen as a poly-ribbon, since the glues forming the polyominoes appear only once and guarantee that only polyominoes can be formed. A T_μ-ribbon may have up to two incomplete polyominoes at the extremities, but if we discard them, then it can be represented as a unique poly-ribbon, and hence as the representation of a unique (T, N_{vN})-zipper. Let $\mathcal{R}$ be the equivalence relation which states that two T_μ-ribbons are equivalent if they represent the same (T, N_{vN})-zipper. Then, the simulation ψ is a bijection between the set of (T, N_{vN})-zippers $\mathfrak{Z}_{T,N_{vN}}$ and the set of T_μ-ribbons quotiented by $\mathcal{R}$, $\mathfrak{R}_{T_\mu/\mathcal{R}}$.

In addition, the polyominoes used in the above simulation are rectilinear polyominoes, i.e., a simple sequence of tiles outlining a shape. These are a particular case of general polyominoes, making the simulation more powerful.

The simulation of a zipper in Moore neighborhood N_M (i.e., adding diagonal communications) is more complex, because diagonal glues cross, as illustrated in Fig. 4.

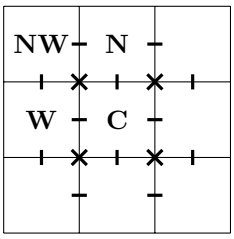
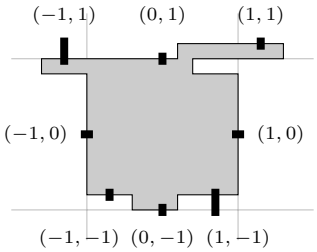


Figure 4: Communication between tiles in Moore neighborhood.

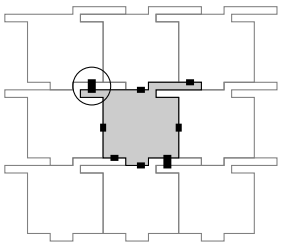
The consequence is that diagonal bumps would also have to cross each other. This issue is solved in [10] using a method that we summarize here, because it will turn out to be useful in our constructions.

Theorem 3.2 ([10]). *Let T be a tile system in neighborhood N_M, with set of glues X. There exist a simulation ψ and a tile system T_μ in neighborhood N_{vN}, with set of glues X_μ, such that any (T, N_M)-zipper can be simulated by a T_μ-ribbon.*

Proof. The global idea of the simulation is the same as for the proof of Theorem 3.1, we need to define a function φ for all tiles from T. In Figs. 5(a) and 5(b) we present a picture of the shape that can be used to simulate an initial tile of T. It differs slightly from the shape described in [10], but the idea is the same and this new shape is an introduction to our results.



(a) A single shape. Next to the communication places is written the neighbor it communicates with.

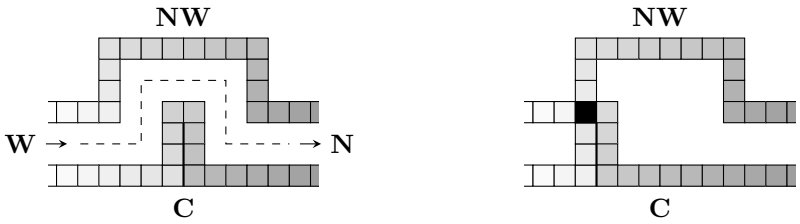


(b) The shape and its 8 neighbors.

Figure 5: Shape used to simulate a tile in Moore neighborhood, filled with gray for clarity sake. The communication bumps and dents are represented by short thick lines.

This shape is turned into a polyomino using new microtiles from T_μ, taking into account the input and output directions. Then, all but one of the communications can be done as previously, by modifying the position of bumps and dents to simulate different glues. The only issue is the communications between the Northwest and Southeast neighbors (circled on Fig. 5(b)). We have to be

able to relay information about these diagonal glues without the physical touch between edges of tiles. According to the notation from Fig. 4, we need to check the glue compatibility between the central tile **C** and its Northwest neighbor **NW**, as well as between the tiles **N** and **W**. This can be accomplished by a geometrical construction such as the one in Figs. 6(a) and 6(b).



(a) Glues matching between **C** and **NW**. (b) Glues not matching: the ribbons overlap.

Figure 6: "Crossing" of information. The top ribbon segment is part of **NW**, the bottom ribbon segment is part of **C**, and the space in between can be filled by 2 layers of microtiles for **W** to actually reach **N**.

This construction checks the match between the West and North tiles in the old-fashioned way, by physically matching the adjacent bump and dent of the corresponding polyominoes. The novelty of this construction is that the glue-match between the central and Northwest tiles is accomplished without the respective polyominoes ever touching. Moreover, this construction works even in the case of partial tilings where the West tile might be missing. This is accomplished by carefully designing the shape and space between the bump and the dent so that, whether or not the tile between these polyominoes is present, their shapes will be compatible and not overlap if and only if the glues of the corresponding tiles were compatible. In order to achieve this and cross the 2 layers of microtiles forming the polyomino **W**, the bump of **C** should be 3 microtiles high, and the dent of **NW** should be 8 microtiles wide and 3 microtiles deep. In the sequel, the inner layers of **W** are called *bridges*.

For any tile $t \in T$, if a polyomino in the set $\varphi(t)$ has a bridge, then $\varphi(t)$ should contain the same polyomino with the bridge at all possible locations, matching all possible glues. Indeed, the bridges do not participate in a communication, they are just a kind of "relay" which should be able to match any glue. Therefore, $\varphi(t)$ contains $|X|$ copies of each of the 12 polyominoes given by the above construction, for the $|X|$ possible locations of the bridges (one per glue). The end of the proof is similar to the proof of Theorem 3.1 and is left to the reader. □

Remark 3.2. A simpler construction [2] using "pitcher-tiles" (see Fig. 7) solves the problem of information crossing when simulating Moore-neighborhood zippers by ribbons, but only when the zipper is a *total* tiling. In that case, one could use (with the notation from Fig. 7) the spike of the **C** polyomino which

conveys information to **NE** as "information carrier" to transmit information from **E** to **N**. This construction does not work as such in the case of zippers which are partial tilings, because the "carrier" **C** tile might be altogether absent.

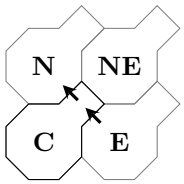


Figure 7: "Pitcher-tiles" used to relay diagonal information, for total-tiling zippers only.

3.2. Simulating Arbitrary Linear Neighborhoods

We call *linear* neighborhood a neighborhood N such that if $(i, j) \in N$ then $j = 0$. Although this is a sub-case of the general case studied in the next section, we will explain it in detail since it lays the base of the general study, and it is much simpler to describe and understand. Also note that this result was already announced in [11], but the polyomino used there was not easily generalizable to arbitrary neighborhoods. First, we state an initial remark which allows us to consider only *connected* linear neighborhoods $N_n = \{(i, 0) \in \mathbb{Z}^2 \mid 0 < |i| \leq n\}$.

Remark 3.3. For a given set of glues X, any tile system T defined in linear neighborhood N can be replaced by an equivalent tile system T' in an appropriate connected linear neighborhood N_n. Indeed, let n be such that $N \subset N_n$, let $g \in X$ be an arbitrary "dummy" glue and $T' = \zeta_g(T)$ a tile system in neighborhood N_n, where $\zeta_g : T \to T'$ is defined for all $t \in T$ and $(i, j) \in N_n$ by

$$\zeta_g(t)_{i,j} = \begin{cases} t_{i,j} & \text{if } (i, j) \in N, \\ g & \text{otherwise.} \end{cases}$$

Then, clearly, $\tau : D \to T$ is a (T, N)-tiling if and only if $\tau' : D \to T'$ defined by $\tau'(x, y) = \zeta_g(\tau(x, y))$ is a (T', N_n)-tiling.

First, we state a lemma which generalizes to an arbitrary number of inner layers the crossing operation detailed in the proof of Theorem 3.2.

Lemma 3.3. *In order to fit a bump and a dent spaced by k layers, the bump must be $k+1$ microtiles high, the dent must be $2k+4$ microtiles wide and $k+1$ microtiles deep.*

Proof. The result is obtained by an immediate generalization of Fig. 6(a) to k white layers instead of 2. □

We now prove the main result of this section for neighborhoods N_n.

Theorem 3.4. *Let T be a tile system in connected linear neighborhood N_n, with set of glues X. There exist a simulation ψ and a tile system T_μ in neighborhood N_{vN}, with set of glues X_μ, such that any (T, N_n)-zipper can be simulated by a T_μ-ribbon.*

Proof. As previously, we are going to replace a tile of the (T, N_n)-zipper by a shape, leading to a set of polyominoes. The general shape of the polyominoes is illustrated in Fig. 8. It consists of a spike sent from the original square to the neighbor at distance n.

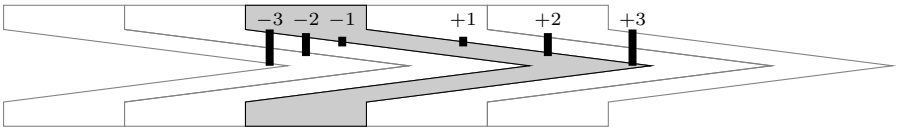


Figure 8: General shape simulating a tile in arbitrary linear neighborhood. In this example, the neighborhood is $N_3 = \{(-3,0), (-2,0), (-1,0), (1,0), (2,0), (3,0)\}$. The shape is drawn in thin black, and the vertical thick lines are the communication places with the neighbor whose abscissa is the number indicated above it. These neighbors are drawn in gray.

As shown on the picture, communication bumps can be put on the spike, while dents are located inside the initial square. For matching the glues between one tile and its neighbor $(i, 0)$, the bump will cross $i-1$ other spikes, we will see later how many layers of microtiles it represents. Indeed, the essential part of the simulation is the discretization of this shape into a polyomino of microtiles. A final polyomino is represented in Fig. 9.

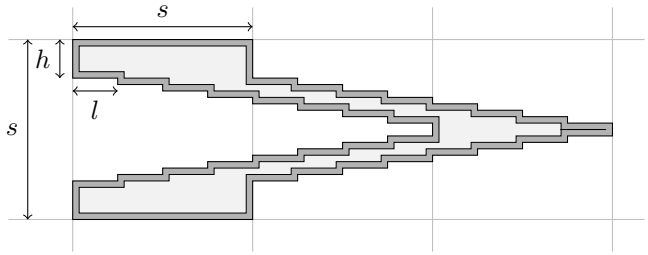


Figure 9: Detail of the polyomino used for arbitrary linear-neighborhood zippers. Here, $n = 2$, $l = 7$, $s = 28$ and $h = 6$. The dark gray contour is filled by microtiles, the light gray area is filled for clarity and does not necessarily contain microtiles.

We define the following variables. The height of the polyomino is s microtiles, it is also the width of the initial tile and therefore the scale of the final poly-ribbon. The vertical space between the top of the polyomino and the beginning of the spike is denoted h, and since the spike is centered, the space between the end of the spike and the bottom is also h. Finally, the spike is a succession of horizontal segments of l microtiles, thus the slope of the spike is $\pm 1/l$. Then, the following constraints have to be respected when constructing the polyomino.

- The integer l has to be as large as necessary, leaving enough horizontal space for crossings, as suggested by Lemma 3.3. Similarly, h needs to provide enough vertical space for the dents. Formally, $l \geq \alpha$ and $h \geq \beta$, where α and β will be given later on.

- The polyomino must be at least 3 layers wide everywhere, two for the inner and outer layers of the spike, and one for a potential junction of the path from the input microtile to the output microtile (as in Fig. 3(c)). As a consequence, $h \geq 3$ and $s \geq 3l + 3$, since the inner layer of the spike must go down by 3 before reaching $s - 3$. The first constraint can be removed (assuming $\beta \geq 3$), while the second one can be replaced by $s \geq 4l$ (provided $l \geq 3$, which is the case if $\alpha \geq 3$), so that s can be exactly divided in 4 horizontal segments everywhere on the spike.

- The initial tile being a square, the height is s and can also be written $h + 2\lfloor ns/l \rfloor + h$ (the spike goes ns microtiles to the right at slope $1/l$). Therefore, after replacing both s by $4l$, we have $h = 2l - 4n$.

Since $h \geq \beta$, because of the last equation l has to be greater than $2n + \beta/2$. For given $n, \alpha, \beta \in \mathbb{N}$, a solution of the system is then

$$\begin{cases} l = \lceil \max(\alpha, 2n + \beta/2) \rceil \\ s = 4l \\ h = 2l - 4n \ . \end{cases}$$

It is quite obvious that translated copies of this polyomino tile the plane with no overlaps, allowing to replace a grid of tiles by a grid of polyominoes. A formal proof of this fact could be made using the characterization of polyominoes tiling the plane given in [6].

We now give some details on how the communications take place. For a more convenient description, we split the polyomino into $n + 1$ horizontal parts of width $s = 4l$, we denote them from left to right by part 0 (which corresponds to the initial square tile) to part n (end of the spike). Each of these parts is divided into 4 horizontal segments of length l, denoted segment 1 to segment 4. As suggested in Fig. 8, the bumps and bridges are located on the horizontal segments on the top of the spike in parts 1 to n, while the corresponding dents are on the horizontal steps on the top of the hole in part 0. For every $0 < i \leq n$, the glue $t_{i,0}$ of the initial tile is allocated some space in every part, on one of the four segments. A simple way to do this is to allocate glue $t_{i,0}$ to segment $(i - 1 \mod 4)$, hence each segment is used for $\lfloor n/4 \rfloor$ or $\lceil n/4 \rceil$ glues. This space is then used in part 0 for a dent, in parts 1 to $i - 1$ for bridges, and in part i for the bump.

Moreover, there are $i - 1$ spikes to cross by the bump encoding $t_{i,0}$. Each spike is 4 layers thick, hence there are at most $4(n - 1)$ layers to cross. This number is fixed, so we can apply Lemma 3.3 with $k = 4(n - 1)$: each glue needs a horizontal space of $(8(n - 1) + 4)$ (width of a dent) + $(|X| - 1)(4(n - 1) + 1)$ (spacing between different possible glues) microtiles. This gives a lower

bound to l, which has to be greater than $\lceil n/4 \rceil \times (|X|\,(4n-3)+4n-1)$. After adding 6 microtiles to separate the dents from the sides of the polyomino, we define

$$\alpha = \lceil n/4 \rceil \times (|X|\,(4n-3)+4n-1)+6 \ ,$$

as the lower bound for l used previously. On the other hand, the depth of the dents is $k+1 = 4n-3$. Consequently, we have another constraint on h which must be greater than $\beta = 4n$ (space for the biggest dent plus the three original layers). This means l greater than $4n$, which is already the case because $l \geq \alpha \geq 4n$. Then,

$$\begin{cases} l = \lceil n/4 \rceil \times (|X|\,(4n-3)+4n-1)+6 \\ s = 4l \\ h = 2l-4n \ . \end{cases}$$

This ensures that our polyomino can be constructed, using an appropriate set of microtiles which will generate the T_μ-ribbon we described. The last step of the construction is the positioning of the input and output microtiles. We can choose to place them at top-left position (path coming from or going to the West), top-right (path from or to the East), middle of the top side (path from or to the North), middle of the bottom side (path from or to the South). Since s is even, the middle of a side is chosen after $\lfloor s/2 \rfloor$ microtiles. Then the inner layer we preserved can be used for joining the input and the output easily; for example, starting from the input microtile, one can draw the contour of the path from the left, just before reaching the output microtile, the path goes one layer inside and goes back to the input microtile where it draws the contour from the right (see Fig. 10 for examples).

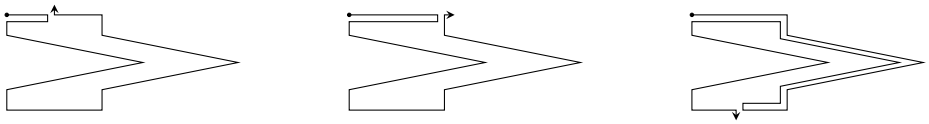


Figure 10: The three different paths when the input is coming from the West.

Finally, putting the appropriate polyominoes one after the other generates a poly-ribbon which is in fact a T_μ-ribbon, it does not overlap if and only if the glues from the initial (T, N_n)-zipper match everywhere. □

We now give results on the "size" of this simulation, which underline the fact that the generated polyominoes can be very complex.

Lemma 3.5. *A polyomino used to simulate a tile of a (T, N_n)-zipper contains $B = \mathcal{O}(n^2)$ bridges.*

Proof. When $n = 2$, there is one bridge between the neighbors at $(-1, 0)$ and at $(1, 0)$; when $n = 3$, there are in addition two bridges between neighbors $(-2, 0)$ and $(1, 0)$, and $(-1, 0)$ and $(2, 0)$. In general, there are $n-1$ bridges

between $(i-n,0)$ and $(i,0)$ for $1 \leq i \leq n-1$, plus $n-2$ bridges between $(i-n-1,0)$ and $(i+n-1,0)$ for $1 \leq i \leq n-2$, and so on. Hence there are $B = \sum_{i=1}^{n-1} i = \frac{1}{2}n(n-1)$ bridges. □

Lemma 3.6. *A polyomino used to simulate a tile of a (T, N_n)-zipper is constituted by $\mathcal{O}(|X|\, n^3)$ microtiles.*

Proof. Drawing the contour of a shape requires:

- s microtiles for the 2 horizontal segments in part 1;
- $4h$ microtiles for all 4 vertical portions;
- $4(ns + 4n)$ microtiles for the 2 spikes (ns for the horizontal segments, $4n$ for the steps down and up);
- at most $s + 2h + 2(ns + 4n) + s/2$ microtiles for joining input and output (worst case when joining left to bottom);
- $2n$ bumps and dents of height at most $\mathcal{O}(n)$ microtiles;
- $B = \mathcal{O}(n^2)$ (Lemma 3.5) bridges of height at most $\mathcal{O}(n)$ microtiles, each one at most 3 layers thick.

Since $s = 4l$ and we can choose $l = \lceil \alpha \rceil = \mathcal{O}(|X|\, n^2)$, after summing all of the above we obtain the result. □

Lemma 3.7. *Every tile of the initial (T, N_n)-zipper is simulated by $\mathcal{O}(|X|\, n^2)$ different polyominoes.*

Proof. For each tile $t \in T$, there are 4 (number of input positions) $\times$ 3 (number of output positions) $\times\, B\,|X|$ (number of different possible bridges) different paths, where B is the number of bridges on the path. Since $B = \mathcal{O}(n^2)$ (Lemma 3.5), there are $\mathcal{O}(|X|\, n^2)$ different polyominoes for one tile when $n \geq 2$. When $n = 1$, there are no bridges and there are only 12 different polyominoes. □

Proposition 3.8. *A T_μ-ribbon simulating a (T, N_n)-zipper needs $|T_\mu| = \mathcal{O}(|T| \cdot |X|^2\, n^5)$ microtiles and $|X_\mu| = \mathcal{O}(|T| \cdot |X|^2\, n^5)$ glues.*

Proof. A (T, N_n)-zipper can use at most $|T|$ tiles, each of them simulated by $\mathcal{O}(|X|\, n^2)$ different polyominoes (Lemma 3.7) constituted by $\mathcal{O}(|X|\, n^3)$ unique microtiles (Lemma 3.6). Hence the simulation needs $|T_\mu| = \mathcal{O}(|T| \cdot |X|^2\, n^5)$ different microtiles.

Since a polyomino is a ribbon which has to be uniquely built, microtiles (except for the input and output ones) have 2 glues which can be found only on one other microtile. Therefore each of these microtiles introduce a new glue, and reuse another: there are at least $|X_\mu| = \mathcal{O}(|T_\mu|) = \mathcal{O}(|T| \cdot |X|^2\, n^5)$ glues. The other two glues are $null_1$ and $null_2$, and the input and output glues microtiles also use glues from X, which does not change the order of magnitude of $|X_\mu|$. □

Remark that if T contains all possible tiles, then $|T| = |X|^{|N_n|} = |X|^{2n}$. Although in general T will contain a much smaller amount of tiles, we can not claim that the number of microtiles and glues used to simulate a (T, N_n)-zipper is polynomial in n.

3.3. Simulating Arbitrary Neighborhoods

In this section, we prove the first of the two main results of this paper, namely that zippers in any neighborhood can be simulated by ribbons (Corollary 3.10). First, note that in a similar way to Remark 3.3, any neighborhood N can be replaced by an equivalent *rectangular* neighborhood $N_{m,n} = \{(i, j) \in \mathbb{Z}^2 \mid 0 \leq |i| \leq m,\ 0 \leq |j| \leq n \text{ and } (i, j) \neq (0, 0)\}$ containing N.

Theorem 3.9. *Let T be a tile system in rectangular neighborhood $N_{m,n}$, with set of glues X. There exist a simulation ψ and a tile system T_μ in neighborhood N_{vN}, with set of glues X_μ, such that any $(T, N_{m,n})$-zipper can be simulated by a T_μ-ribbon.*

Proof. The key of the proof is the generalization of the φ simulation from the proof of Theorem 3.4 to rectangular neighborhoods. The idea is to have a vertical succession of $n + 1$ spikes of length m, each of them being a "sheath" for the next one (Fig. 11).

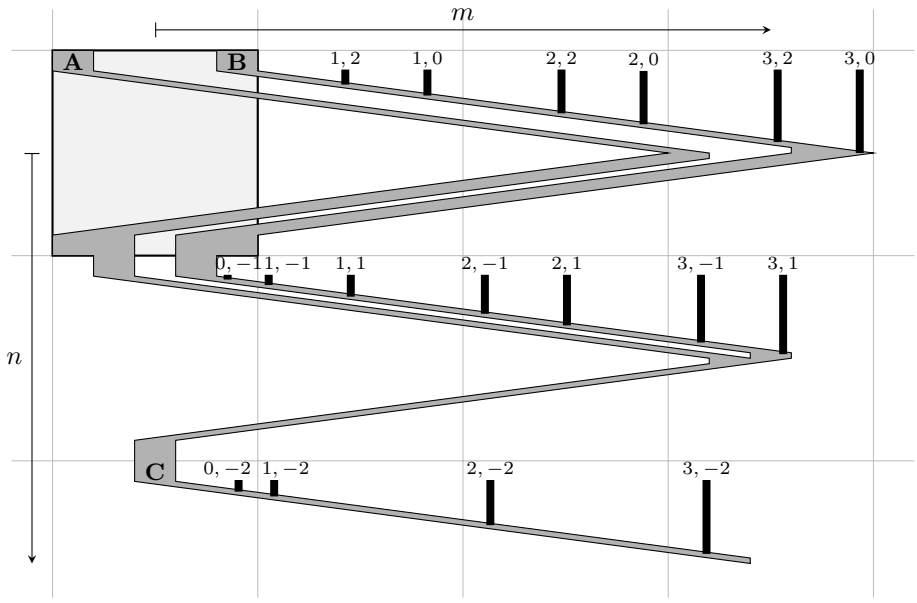


Figure 11: Shape simulating a tile defined in a rectangular neighborhood $N_{m,n}$ of size $(2m + 1) \times (2n + 1)$ (here $m = 3$ and $n = 2$). The initial tile is the light gray square, while the path is filled with darker gray.

The dents used for communications are put all along the spikes, at the places indicated on the picture. The difficulty is to find places for the dents, so that they can be contained in a place where enough space can be reserved. This is achieved as follows.

- North (from $(0, j)$, for $0 < j \leq n$) and West (from $(i, 0)$, for $-m \leq i < 0$) communications are received in the area marked **B** on the picture. This works very similarly to the linear-neighborhood case.
- Northwest communications (from (i, j), for $-m \leq i < 0$ and $0 < j \leq n$) are received in **A**. Again, it is not difficult to see how the information crosses the layers.
- Southwest communications (from (i, j), for $-m \leq i < 0$ and $-n \leq j < 0$) are received in **C**. This is slightly more complicated to understand, since these dents are not located inside the initial square. Putting the dent at the bottom of the polyomino allows to simulate the Southwest communications by Northwest communications, which is then easily achieved by positioning bumps on the spike.

The key point is that the areas marked **A**, **B**, and **C** are scalable, both vertically and horizontally, so they can be made as big as needed for the dents. Indeed, we can denote as previously by s the width and height of the initial tile, hence the scale of the poly-ribbon. Let x be the width of **A**, **B**, **C** and of all the other horizontal subdivisions of s. Since there are two of these blocks for each of the first n vertical spikes, plus one for the last spike, it holds that $s = (2n+1)x$. As in Theorem 3.4, we need the spike to be 3 layers wide, hence the slope $1/l$ of the spikes is defined by l such that $x \geq 3l + 3$. Again it is possible to decide that $x = 4l$, i.e., a block is made of four descending horizontal segments. Finally, let h be the height of the **A**-**B**-**C** blocks. The fact that the initial tile is a square is expressed by $s = 2h + 2\lfloor ms/l \rfloor$. We have the following equations:

$$\begin{cases} x = 4l \\ s = (2n+1)x \\ s = 2h + 2\lfloor ms/l \rfloor \end{cases}.$$

Besides, to ensure enough space for the bumps and dents, we want to make sure that x and h are as big as necessary, i.e., $x \geq \alpha$ and $h \geq \beta$ (the exact values of α and β will be given later). Once solved, the system gives $h = (2n+1)(2l-4m)$. Since h should be greater than β and x greater than α, l needs to be greater than $\max(\alpha/4, \beta/(4n+2) + 2m)$. Thus, a solution to the system which guarantees that the polyomino can be constructed is

$$\begin{cases} l = \lceil \max(\alpha/4, \beta/(4n+2) + 2m) \rceil \\ x = 4l \\ h = (2n+1)(2l-4m) \\ s = 4(2n+1)l \end{cases}.$$

A last technical difficulty is the description of how the spikes shrink to fit into the previous one. The general way to do this is illustrated on Fig. 12, which zooms on the rightmost part of the first spike of a polyomino.

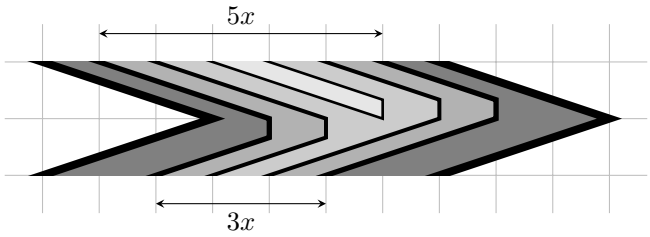


Figure 12: Illustration of how spikes shrink from width ax to $(a-2)x$. The darkest gray corresponds to the spike of the initial tile, lighter grays represent the spike of farther North neighbors.

The outer spike does not shrink because it does not need to, since the shrinking will happen at the bottom of the initial tile (see Fig. 11). This is not necessary, but it allows a tiling of the plane without any holes between polyominoes. The other spikes shrink by x microtiles on the left and on the right by going down 4 tiles (remember that the slope of the spikes is $1/l = 4/x$). Remark the slight asymmetry of the shrinking, which has to take place after reaching height $s/2$ on the left, and just before on the right. With all these conditions respected, it should be clear that vertically and horizontally translated copies of this polyomino tile the plane with no overlaps.

It remains to determine the bounds α and β. An application of Lemma 3.3 to all the spikes and dents gives us the maximal size of bumps and dents. It is obtained for the diagonal communication between a tile and its neighbor located at $(m,-n)$, which crosses $n+(m-1)(2n+1)$ spikes of thickness 4 layers, hence a total number of $k = 4(2mn+m-n-1)$ layers. It follows that $\beta = k+3$ to ensure a free layer between the top of a dent and the upper side of the polyomino. The bound α is more complicated to express. As for Theorem 3.4, a communication bump-dent needs $2k+4$ horizontal space, plus an extra $(|X|-1)(k+1)$ microtiles for the different glues. The size x has to allow $m+n$ dents in $\mathbf{B}$, mn dents in $\mathbf{A}$ and $\mathbf{C}$. Hence, after adding 6 microtiles for preserving the borders of the block, $\alpha = \max(mn, m+n)\times(2k+4+(|X|-1)(k+1))+6$, with $k = 4(2mn+m-n-1)$.

The rest of the proof (positioning and joining the input and output microtiles, bijection between a $(T, N_{m,n})$-zipper and a set of unique poly-ribbons) is unchanged from the proof of Theorem 3.4. □

Corollary 3.10. *Let T be a tile system in arbitrary neighborhood N, with set of glues X. There exist a simulation ψ and a tile system T_μ in neighborhood N_{vN}, with set of glues X_μ, such that any (T,N)-zipper can be simulated by a T_μ-ribbon.*

Remark 3.1 can be extended to the general case, hence any T_μ-ribbon represents a unique (T,N)-zipper. Also note that a result similar to Proposition 3.8

could be stated, but it would be more complex and of little interest since the number of tiles would be a lot bigger.

To illustrate the achievability (and the complexity) of this construction, Fig. 13 gives a complete example of a polyomino for the simulation of a linear-neighborhood (T, N_2)-zipper, with $|X| = 2$.

4. Extension to Arbitrary-Neighborhood Tilings

We now explain how the above construction can be modified, in order to simulate arbitrary-neighborhood tilings by von Neumann-neighborhood poly-tilings, to obtain our second main result (Corollary 4.3). Before that, we describe a simpler construction used in [3], which can be adapted to do such a simulation, but only in the case of total tilings. We finally study some properties of the tilings which are preserved by our simulation technique.

4.1. Total Tilings Constructions

In [3], the authors use a construction which can be adapted to the simulation of total tilings. Their idea is to replace each arbitrary-neighborhood tiles by von Neumann-neighborhood tiles, these new tiles (called *supertiles* to avoid misunderstandings) being groupings of several of the initial tiles, such that supertiles group all the tiles which belong to the neighborhood of the initial tile. This leads to the following statement.

Proposition 4.1. *Let T be a tile system in rectangular neighborhood $N_{m,n}$, with set of glues X. There exist a simulation ψ and a tile system T' in neighborhood N_{vN}, with set of glues X', such that any total $(T, N_{m,n})$-tiling can be simulated by a total (T', N_{vN})-tiling.*

Proof. Let τ be a $(T, N_{m,n})$-tiling. We replace tiles from T by new tiles from $T' = T^{(2m+1)(2n+1)}$ called *supertiles*, each of them encoding $(2m+1) \times (2n+1)$ tiles. The 4 von Neumann glues are $(m+1) \times (2n+1)$ (East and West) or $(2m+1) \times (n+1)$ (North and South) rectangles. Therefore, two supertiles stick if the matching glues encode the same tiles, allowing to transmit information to distant neighbors. Figure 14 illustrates this construction in the case of Moore neighborhood, i.e., $m = n = 1$. In this figure, we use the following notation: for a tile $t \in T$ at position $(x, y) \in \mathbb{Z}^2$, $N(t)$ [resp., $S(t)$, $E(t)$, $W(t)$] represents the tile $t' \in T$ located at position $(x, y+1)$ [resp., $(x, y-1)$, $(x+1, y)$, $(x-1, y)$]; and denote $XY = X \circ Y$ the composition of any of these functions $X, Y \in \{N, S, E, W\}$.

In this case, the polyominoes simulating the original tiles are made of only one supertile. Besides, for each tile $t \in T$, the set $\varphi(t)$ contains all the one-tile tilings consisting of a supertile centered on t, with all the admissible neighbors of t. Any total tiling $\tau \in \mathfrak{T}_{T,N_{m,n}}$ is then simulated by an appropriate poly-tiling $\psi(\tau)$, seen as a total (T', N_{vN})-tiling $\tau' \in \mathfrak{T}_{T',N_{vN}}$. □

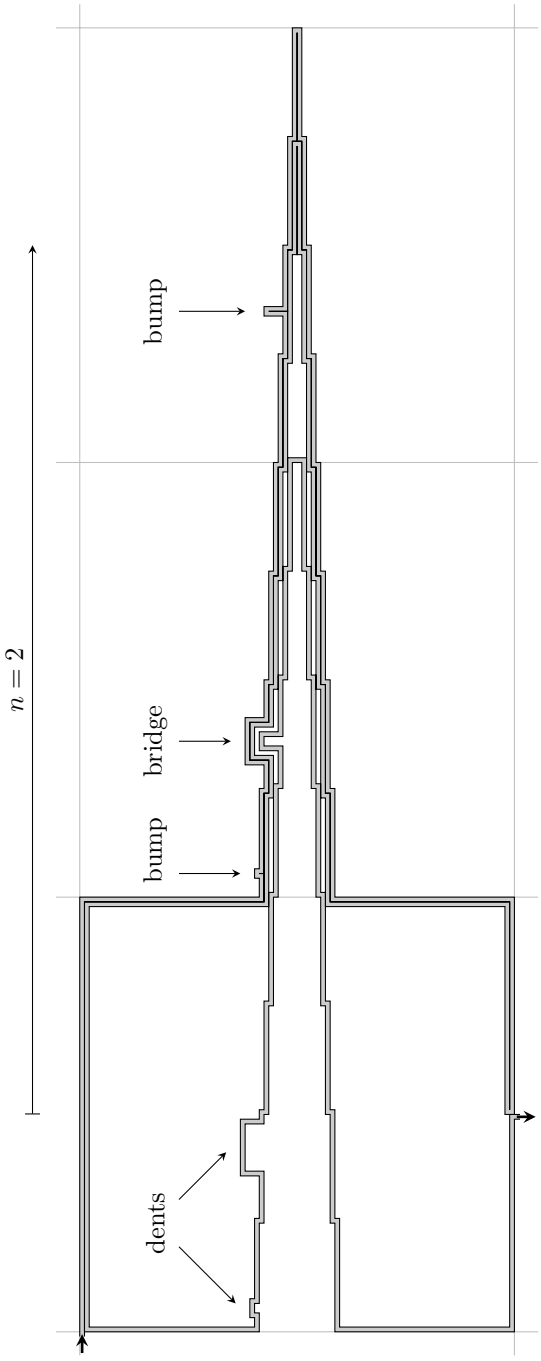


Figure 13: Rotated polyomino simulating a tile of a (T, N_2)-zipper, with $|X| = 2$, $l = 23$, $L = 92$, $h = 38$, input on the left and output at the bottom. The microtiles are located in the gray layer, the input and output are indicated by the black arrows.

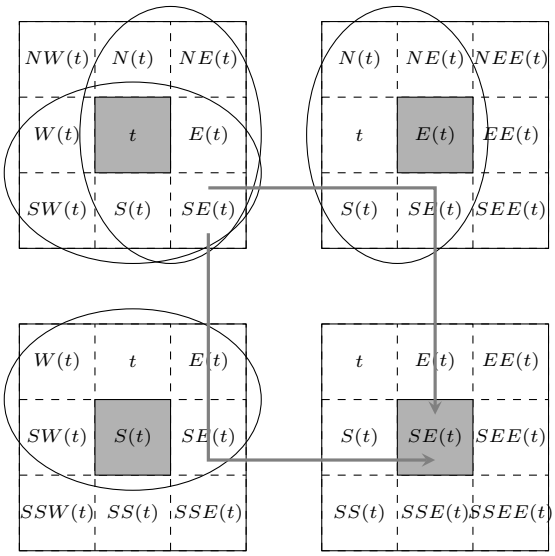


Figure 14: Supertiles assembly for simulating neighborhood $N_M = N_{1,1}$. The glues consist of the circled elements, they are used to transmit the information diagonally, as indicated by the arrows.

Remark that this technique can not be applied directly to the simulation of total zippers. Indeed, one would need to be able to transfer an arbitrary amount of information when simulating zippers such as the one represented in Fig. 15, since this amount would depend on the length of the zipper before it goes back.

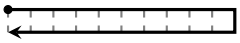


Figure 15: Von Neumann-neighborhood zipper, going backwards. The longer it is, the more information (represented as dashed gray lines) would need to be encoded in the supertiles.

However, Theorem 3.1 allows to transfer information on a ribbon in all 4 directions. This result, in conjunction with Proposition 4.1, allows a simpler construction in the case of arbitrary-neighborhood total zippers, by first reducing to a von Neumann-neighborhood zipper with the supertiles technique, and then simulating it by a two-tile-neighborhood ribbon. This idea is similar to the one suggested in Remark 3.2, since it uses intermediate polyominoes to relay information. The main difference is that the polyomino-based simulation is more general and can be used for arbitrary neighborhoods, while pitcher-tiles were designed specifically for the Moore neighborhood.

Also note that this simulation does not work for partial tilings, since in the absence of intermediate supertiles the information can not be transmitted. For

example, in Fig. 14, if the top-right and bottom-left supertiles are omitted, it is not possible to ensure that the bottom-right tile is centered on $SE(t)$.

This issue can be partially solved [25], by introducing a new symbol "no tile" in the supertiles, when a tile is absent. In this case, there will be supertiles at each position of the plane (possibly consisting only of symbols "no tile"). However, this would cause every partial tiling to be simulated by a total tiling, which is not suitable in most applications.

4.2. Polyomino Construction

The construction described in Sect. 3 transforms a tile into a polyomino of microtiles, keeping the path unchanged at the macro-level. Hence, a similar construction can be used to prove another result, which states that arbitrary-neighborhood tilings can be simulated by von Neumann-neighborhood tilings. As usual, we first prove the result for rectangular neighborhoods, and deduce the general case from that.

Theorem 4.2. *Let T be a tile system in rectangular neighborhood $N_{m,n}$, with set of glues X. There exist a simulation ψ and a tile system T_μ in neighborhood N_{vN}, with set of glues X_μ, such that any $(T, N_{m,n})$-tiling can be simulated by a (T_μ, N_{vN})-tiling.*

Proof. Only a few modifications have to be done to the general polyomino represented in Fig. 11, in the construction of Theorem 3.9, to transform the T_μ-ribbon into a (T_μ, N_{vN})-tiling. First, the inner layer used to join the input and output microtiles can be removed, allowing thinner spikes.

The main difference is that now, all 4 glues of the microtiles have to match with their neighbors, if present. The glues towards the inside of the polyomino (i.e., the glues used for adjacent microtiles from the same polyomino) can be chosen as previously, uniquely for each tile of each polyomino. For the glues used to match with other polyominoes, things are slightly more complex, since they will be used to enforce the correct position of the adjacent polyomino, to avoid misalignment. These glues can be chosen as pairs $(x, y) \in \mathbb{N}^2$. For North and East glues, (x, y) represents the position of the microtile inside the polyomino, counting from the bottom-left part of the polyomino for example (hence, the microtile at the top-left of the polyomino will have glue $(1, s(n + 1))$ on the North side). For South and West glues, this pair should be the position of the microtile on the neighboring polyomino, to ensure proper match. Therefore, the West glue of the top-left microtile will be $(s, s(n + 1))$, since it has to match with the top-right tile of the polyomino adjacent on the right. Note that $1 \leq x \leq s(m + 1)$ and $1 \leq y \leq s(n + 1)$ (see Fig. 11), therefore the number of these glues is bounded, once T and $N_{m,n}$ are fixed. These glue do not need to encode any information from the original set of glues X, since this matching will be enforced by the shape of the polyominoes. Their role is only to align properly the polyominoes.

Then, each $(T, N_{m,n})$-tiling is associated with a set of (T_μ, N_{vN})-tilings by this construction. Conversely, for any poly-tiling using the tile system T_μ in

neighborhood N_{vN}, one can discard the incomplete polyominoes and use the remaining ones to recover the initial tiles at their respective positions, with the help of the function φ^{-1} applied to each of the complete polyominoes. □

As explained at the beginning of Sect. 3.3, any neighborhood can be replaced by an equivalent rectangular neighborhood, hence the following corollary.

Corollary 4.3. *Let T be a tile system in arbitrary neighborhood N, with set of glues X. There exist a simulation ψ and a tile system T_μ in neighborhood N_{vN}, with set of glues X_μ, such that any (T, N)-tiling can be simulated by a (T_μ, N_{vN})-tiling.*

Remark that while the construction of Theorem 3.9 ensures that every T_μ-ribbon is a poly-ribbon (Remark 3.1), with the construction of Theorem 4.2, every (T_μ, N_{vN})-tiling is not necessarily a poly-tiling. Indeed, the choice of the glues from X_μ still guarantees that a (T_μ, N_{vN})-tiling consists only of (partial) polyominoes, but the alignment on a grid of these polyominoes is not ensured. This is because the (T_μ, N_{vN})-tiling may not be connected, in which case it is not possible to force the correct positioning of these unconnected components, as it was done for the simulation by a (necessarily connected) ribbon. However, since this construction is mainly proposed for self-assembly situations, this should not be an issue. Indeed, a tiling obtained by self-assembly will grow increasingly from an initial "seed" tile, and will thus be constantly connected, in which case the alignment of the polyominoes can be enforced by the choice of the glues. Besides, every tiling can be simulated by a connected (T_μ, N_{vN})-tiling, provided the different components are connected by incomplete polyominoes with no glue information. After discarding these artificial "links", an aligned poly-tiling remains which simulates the original tiling.

Thus, the equivalence now becomes "every *connected* (T_μ, N_{vN})-tiling represents a unique (not necessarily connected) (T, N)-tiling", and ψ is a bijection from $\mathfrak{T}_{T,N}$ to the set of connected (T_μ, N_{vN})-tilings quotiented by an appropriate equivalence relation.

4.3. Properties Preserved by the Simulation

The main interest of this construction is that it guarantees that some properties of the initial tiling are preserved, and still verified in the final poly-tiling. For example, unlike what happens with the simple construction explained in Sect. 4.1, a partial (T, N)-tiling is simulated by a partial (T_μ, N_{vN})-tiling. The converse is also true, as well as some other important properties like periodicity and convexity, as explained in this section.

Partial tilings. Clearly, from our construction, a partial tiling is simulated by partial tilings. For the converse, remark that our construction creates "holes" in a poly-tiling, because the polyominoes we build are hollow. Then, provided we fill the polyominoes by new tiles with unique glues, we obtain the following immediate result.

Theorem 4.4. *Let T be a tile system in arbitrary neighborhood N, with set of glues X. There exists a simulation ψ such that every (T, N)-tiling τ is total if and only if all (T_μ, N_{vN})-tilings in $\psi(\tau)$ are total.*

Note that when τ is total, $\psi(\tau)$ is a singleton. Indeed, the polyominoes of $\varphi(t)$ differ only by the position of the bridges. When the tiling is total, all bridges are constrained by the neighboring polyominoes, hence only one element of the set $\varphi(t)$ is possible. The final poly-tiling is therefore uniquely defined.

Remark 4.1. Similarly, remark that a tiling is finite [resp., connected] if and only if all the poly-tilings which simulate it are finite [resp., connected]. As a consequence, the simulation of polyominoes is achieved by polyominoes only.

Periodic tilings. Periodicity is an essential notion in the classical tiling theory [8, 18], it should be preserved by a meaningful simulation. The following result shows that this is the case.

Theorem 4.5. *Let T be a tile system in arbitrary neighborhood N, with set of glues X. There exists a simulation ψ such that every (T, N)-tiling τ is periodic if and only if at least one of the (T_μ, N_{vN})-tilings in $\psi(\tau)$ is periodic.*

Proof. Clearly, if a poly-tiling is periodic, then the tiling it simulates is also periodic. Conversely, if a tiling is periodic, then one of the poly-tilings which simulate it has to be periodic. In the case of a total tiling, this fact is obvious. In the case of a partial tiling, it suffices to choose the poly-tiling with the correct bridge constraints at the borders, so that the periodic pattern of the tiling repeats all over. □

Note that if the periods of a tiling τ are $p_v, p_h \in \mathbb{N}_+$, the periods of the periodic poly-tiling in $\psi(\tau)$ are $sp_v, sp_h \in \mathbb{N}_+$, where s is the scale of the poly-tiling.

Convex tilings. The different convexity notions (line, column, or both) are important when dealing with polyominoes [5, 9], which are a particular case of tilings. Our construction does not preserve the convexity of a tiling, but it is possible to modify it in order to preserve line or column convexity.

Theorem 4.6. *Let T be a tile system in arbitrary neighborhood N, with set of glues X. There exists a simulation ψ [resp., ψ'] such that every (T, N)-tiling τ is line convex [resp., column convex] if and only if all (T_μ, N_{vN})-tilings in $\psi(\tau)$ are line convex [resp., column convex].*

Proof. Let us concentrate on the preservation of line convexity, the column convexity can be achieved by rotating the polyominoes by 90°. Our construction needs to be modified, for two reasons breaking line convexity, as shown of Fig. 11: the sheaths in the spikes and the dents create horizontal holes.

First of all, the sheaths can be replaced by stacking layers, as shown on Fig. 16. This requires some technical details to make sure that the polyomino

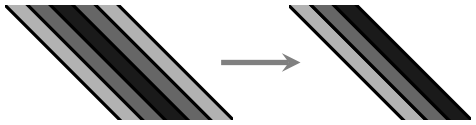


Figure 16: Replacing sheaths by layers. The different polyominoes are represented by different gray levels, they stack instead of interlocking.

remains connected. Besides, the **B** block (Fig. 11) would disappear, so the horizontal communications should arrive in **A** and the vertical ones in **C** (from above).

Replacing the dents by a shape which preserves the line convexity is slightly harder. The idea is to replace the hole by two sets of stairs, using again the position of the stairs to ensure the glue matching, as represented in Fig. 17. In this figure, the integer g is the position of the dent, as induced by some glue. In order for the top and bottom polyominoes to match, the bump had to be located at position $g + k + 1$, where k is the number of layers crossed. In the new construction, the glue matching is enforced by the fact that in the bottom polyomino, if g is too big then there is an overlap on the left stairs, and if g is too small then there is an overlap on the right stairs.

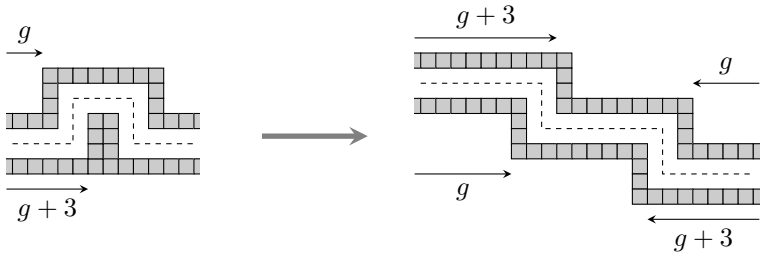


Figure 17: Replacing bumps and dents by two stairs. Here, there are $k = 2$ layers to be crossed.

The problem with this technical step is that crossing k layers implies that the spike goes down by $2(k + 1)$. If γ is the number of communication places in a segment of length l, and δ is the maximum number of layers which can be crossed, then each segment of length l in the spike would go down by $2\gamma(\delta + 1)$ instead of 1. This imposes new constraints on h and s, but the discretization system is still solvable, at the additional cost of increasing l by a factor $2\gamma(\delta+1)$.

Then, all polyominoes $\varphi(t)$ are line convex, so replacing each tile t from a line convex tiling by one of these polyominoes generates a line convex poly-tiling. The converse is trivial, if a poly-tiling is line convex then it simulates a line convex tiling. □

5. Conclusions and Perspectives

In this paper, we proved that any zipper formed with tiles defined in an arbitrarily complex neighborhood can be simulated by a ribbon obtained by the catenation of simple polyominoes. Each of the new microtiles used for the simulation has, when placed on a ribbon, only two neighbors: its predecessor and its successor on the path. A similar construction can be achieved for tilings, in order to replace arbitrary neighborhoods by the simpler von Neumann neighborhood, and at the same time preserve some properties of the tiling.

A few research directions can extend this work. For example, the simulation relies on the fact that zippers and ribbons are not self-crossing. Although this is the standard way to define them, they could be generalized to other, self-crossing, notions. It would be interesting to see if similar results could be obtained in this case, using new constructions. Another interesting topic would be the generalization of this construction to three-dimensional neighborhoods. The spikes could be designed to go in all three dimensions, however, new problems occur and would have to be solved by original techniques.

Another possible improvement is justified by the observation that our constructions are not yet adapted to practical dynamical implementations, such as DNA self-assembly. Theoretically, our constructions could be used for practical purposes: for example, as mentioned in [19], the construction of Theorem 3.1 can be used to simulate a Turing machine at temperature 1 (i.e., DNA tiles attach to the growing assembly whenever one glue matches). But this produces a lot of "garbage" consisting of many blocked polyominoes, for a limited number of correct assemblies. Indeed, there are many dynamical scenarios wherein, for example, the self-assembly of a "bad" polyomino is started, that blocks any further aggregation. This issue could be solved by adding the possibility to recover from a wrong start, for example by allowing tiles to "unstick", as suggested in [24, 1]. Further modifications would then be necessary to guarantee that, also in this setting, our constructions can self-assemble fully and correctly in finite time.

Also remark that in our constructions, we emphasized the regularity of the polyominoes, to provide easier general constructions and aggregation. This led however to a blow-up in the number of new microtiles necessary to simulate a zipper, potentially leading to more experimental issues. The next step would be to optimize the construction according to some of the following criteria: number of polyominoes associated with a tile, number of microtiles appearing in a polyomino, number of glues used to build the polyominoes, etc.

An alternative solution to these two problems would be the use of staged self-assembly [12] for experiments. This formalism would allow to add an initial stage to construct the polyominoes in separate bins, before mixing them in the final solution, preventing the formation of "bad" assemblies. Besides, staged self-assembly would dramatically decrease the number of required glues (hence the number of microtiles), by adding even more initial stages to produce the polyominoes. This would offer more control on the order in which microtiles attach, thus removing the necessity for unique glues. However, this would be

achieved at the cost of increased control by an external operator during the self-assembly process. This trade-off between external control and tile complexity is often encountered in DNA self-assembly (see, e.g., [14, 12]), and is currently being investigated by the authors.

The authors would like to thank Shinnosuke Seki for his useful suggestion simplifying the construction in Fig. 11, and Leonard Adleman, Jarkko Kari, and Erik Winfree for discussions.

References

[1] L. Adleman. Towards a mathematical theory of self-assembly. Technical Report 00-722, Department of Computer Science, University of Southern California, 2000.

[2] L. Adleman. Personal communication, 2001.

[3] L. Adleman, J. Kari, L. Kari, and D. Reishus. On the decidability of self-assembly of infinite ribbons. In *IEEE Symposium on Foundations of Computer Science (FOCS 2002)*, pages 530–537, 2002.

[4] L. Adleman, J. Kari, L. Kari, D. Reishus, and P. Sosik. The undecidability of the infinite ribbon problem: Implications for computing by self-assembly. *SIAM Journal on Computing*, 38(6):2356–2381, 2009.

[5] E. Barcucci, A. D. Lungo, M. Nivat, and R. Pinzani. Reconstructing convex polyominoes from horizontal and vertical projections. *Theoretical Computer Science*, 155(2):321–347, 1996.

[6] D. Beauquier and M. Nivat. On translating one polyomino to tile the plane. *Discrete and Computational Geometry*, 6:575–592, 1991.

[7] F. Becker, E. Rémila, and N. Schabanel. Time optimal self-assembling of 2D and 3D shapes: The case of squares and cubes. In *Preliminary proceedings of International Meeting on DNA Computing (DNA 14)*, pages 78–87, 2008.

[8] R. Berger. The undecidability of the domino problem. *Memoirs of the American Mathematical Society*, 66:1–72, 1966.

[9] M. Chrobak and C. Dürr. Reconstructing HV-convex polyominoes from orthogonal projections. *Information Processing Letters*, 69(6):283–289, 1999.

[10] E. Czeizler and L. Kari. Towards a neighborhood simplification of tile systems: From Moore to quasi linear dependencies. Preprint, 2008.

[11] E. Czeizler and L. Kari. Geometrical tile design for complex neighborhoods. *Frontiers in Computational Neurosciences*, 3:16, 2009.

[12] E. Demaine, M. Demaine, S. Fekete, M. Ishaque, E. Rafalin, R. Schweller, and D. Souvaine. Staged self-assembly: Nanomanufacture of arbitrary shapes with $O(1)$ glues. In *International Meeting on DNA Computing (DNA 13)*, volume 4848 of *Lecture Notes in Computer Science*, pages 1–14, 2008.

[13] K. Fujibayashi, R. Hariadi, S. H. Park, E. Winfree, and S. Murata. Toward reliable algorithmic self-assembly of DNA tiles: A fixed-width cellular automaton pattern. *Nano Letters*, 8(7):1791–1797, 2007.

[14] M.-Y. Kao and R. Schweller. Reducing tile complexity for self-assembly through temperature programming. In *ACM-SIAM Symposium on Discrete Algorithms (SODA 2006)*, pages 571–580, 2006.

[15] J. Kari. Theory of cellular automata: A survey. *Theoretical Computer Science*, 334:3–33, 2005.

[16] L. Kari and B. Masson. Simulating arbitrary-neighborhood tilings by polyominoes. In *Foundations of Nanoscience (FNANO 2009)*, poster, 2009.

[17] C. Mao, T. Labean, J. Reif, and N. Seeman. Logical computation using algorithmic self-assembly of DNA triple-crossover molecules. *Nature*, 407:493–496, 2000.

[18] R. Robinson. Undecidability and nonperiodicity for tilings of the plane. *Inventiones Mathematicae*, 12:177–209, 1971.

[19] P. Rothemund. *Theory and Experiments in Algorithmic Self-Assembly*. PhD thesis, University of Southern California, 2001.

[20] P. Rothemund, N. Papadakis, and E. Winfree. Algorithmic self-assembly of DNA Sierpinski triangles. *PLoS Biology*, 2(12):e424 (13 pages), 2004.

[21] P. Rothemund and E. Winfree. The program-size complexity of self-assembled squares (extended abstract). In *ACM Symposium on Theory of Computing (STOC 2000)*, pages 459–468, 2000.

[22] R. Schulman and E. Winfree. Programmable control of nucleation for algorithmic self-assembly. In *International Workshop on DNA Computing (DNA 10)*, volume 3384 of *Lecture Notes in Computer Science*, pages 319–328, 2005.

[23] H. Wang. Proving theorems by pattern recognition – II. *Bell Systems Technical Journal*, 40:1–42, 1961.

[24] E. Winfree. *Algorithmic Self-Assembly of DNA*. PhD thesis, California Institute of Technology, 1998.

[25] E. Winfree. Personal communication, 2009.

[26] E. Winfree, F. Liu, L. Wenzler, and N. Seeman. Design and self-assembly of two dimensional DNA crystals. *Nature*, 394:539–544, 1998.

[27] E. Winfree, X. Yang, and N. Seeman. Universal computation via self-assembly of DNA: Some theory and experiments. In *DNA-Based Computers II*, volume 44 of *DIMACS Series*, pages 191–213, 1998.