Published in final edited form as:

Nat Comput. 2010 June 1; 9(2): 437–455. doi:10.1007/s11047-009-9157-5.

TRANSDUCER GENERATED ARRAYS OF ROBOTIC NANO-ARMS

EGOR DOLZHENKO¹, NATAŠA JONOSKA¹, and NADRIAN C. SEEMAN²

EGOR DOLZHENKO: EDOLZHEN@MAIL.USF.EDU; NATAŠA JONOSKA: JONOSKA@MATH.USF.EDU; NADRIAN C. SEEMAN: NED.SEEMAN@NYU.EDU

¹UNIVERSITY OF SOUTH FLORIDA, TAMPA, FL 33620, USA

²NEW YORK UNIVERSITY, NEW YORK NY 10003, USA

Abstract

We consider sets of two-dimensional arrays, called here transducer generated languages, obtained by iterative applications of transducers (finite state automata with output). Each transducer generates a set of blocks of symbols such that the bottom row of a block is an input string accepted by the transducer and, by iterative application of the transducer, each row of the block is an output of the transducer on the preceding row. We show how these arrays can be implemented through molecular assembly of triple crossover DNA molecules. Such assembly could serve as a scaffold for arranging molecular robotic arms capable for simultaneous movements. We observe that transducer generated languages define a class of languages which is a proper subclass of recognizable picture languages, but it containing the class of all factorial local two-dimensional languages. By taking the average growth rate of the number of blocks in the language as a measure of its complexity, we further observe that arrays with high complexity patterns can be generated in this way.

Keywords

transducers; finite state automata with output; picture languages; two-dimensional languages; local languages; DNA tiles; robotic arms; DNA arrays

1. Introduction and Background

Synthetic DNA has been designed and shown to assemble into complex species that entail the lateral fusion of DNA double helices, such as DNA double crossover (DX) molecules [8], triple crossover (TX) molecules [22] or paranemic crossover (PX) molecules. Double and triple cross-over molecules have been used as tiles and building blocks for large nanoscale arrays [30, 29], including the Sierinski triangle assembly by DX molecules [28]. Theoretically, it is shown that two-dimensional arrays made of DX or TX DNA can simulate the dynamics of a bounded one-dimensional cellular automaton and so are capable of potentially performing computations as a Universal Turing machine [30]. The essential part in these observations is the natural representation of a Wang tile with a DX or a TX molecule.

Wang tiles have been extensively used in the study of two-dimensional languages and there is a natural correspondence between the sets of blocks assembled with Wang tiles and local two-dimensional languages. The physical representation of Wang tiles with DNA molecules demonstrated recently [29, 30] provides another motivation for studying sets of rectangular blocks of symbols namely two-dimensional languages. It is well known that by iteration of generalized sequential machines (finite state machines mapping symbols into strings) all

computable functions can be simulated (see for ex. [26, 27]). The full computational power depends on the possibility for iterations of a finite state machine. Moreover, there is a natural simulation of iteration of transducers and recursive (computable) functions with Wang tiles [13]. This idea has been developed further in [5] where a successful experimental simulation of a programmable transducer (finite state machine mapping symbols into symbols) with TX DNA molecules having iteration capabilities is reported. This experimental development provides means for generating patterns and variety of two-dimensional arrays at the nano level. Motivated by this recent experimental development here we define a class of two-dimensional languages generated by iteration of transducers, called transducer generated languages and show how physical realization of arrays corresponding to such a language can be used as a platform for arranging robotic nano-arms. An earlier short description of transducer generated languages appears in [7].

The notation and the definitions are included in Section 2 while in Section 3 we show that transducer generated (TG) two-dimensional (picture) languages belong to the class of recognizable (REC) picture languages. We also observe that TG languages include all local (factorial) picture languages, i.e., languages generated by finite sets of Wang tiles. Although there are fairly well developed classifications and theories to study one-dimensional languages, in particular regular languages, the case of two-dimensional languages remains elusive. The class REC defined by A. Restivo and D. Giammarresi [10, 11] was introduced as a natural extension of regular languages in one-dimensional case. They showed that the emptiness problem for these languages is undecidable which implies that many other questions that are easily solved in one-dimensional case become undecidable in two-dimensional case. Recent work by several authors attempts to better understand REC through various approaches such as: design of variants of finite state automata that recognize these languages [1, 2, 17, 18], characterization of determinism of two-dimensional recognizable languages [3, 4], or study of the factor languages of two-dimensional shift spaces [14, 15]. Therefore, the transducer generated languages introduced here define another sub-class of REC languages whose analysis may provide a way to understand some properties of the whole class of REC.

In Section 4, as a measure of complexity, we make some observations about the average growth rate of the number of sub-blocks in a TG picture language. This measure is known in symbolic dynamics as the entropy of the language and it is well understood in the case of one-dimensional regular languages. It is known that in one-dimensional case, the set of entropies of regular languages coincides with the set of logarithms of Perron numbers (see [25]). However, there is no general theory that describes a way to obtain the entropy of a two-dimensional local, or even less a language belonging to the larger class of recognizable languages. It is known that even languages with so called strong transitivity, i.e., mixing properties, have zero entropy [20, 21]. Here, we observe that the value given by a log of a Perron number can be obtained as an entropy value of a transducer generated two-dimensional language. This also shows that the two-dimensional arrays obtained by self-assembly of DNA tiles simulating iterations of a transducer may have rich pattern complexity.

A possible implementation of transducer generated arrays with TX DNA molecules to serve as platforms for arranging robotic nano-arms is presented in Section 6. A PX-JX₂ device introduced in [31], has two distinct positions and each is obtained by addition of a pair of DNA strands that hybridizes with the device such that the molecule is in either PX or in JX₂ position. This device has been assembled sequentially [24] and in an array [6]. In the later case it has been shown that a molecular robotic arm can be attached to the device and controlled simultaneously. The successful molecular implementation of a transducer [5]

forms a basis for assembling transducer generated arrays incorporating molecular robotic arms to be moved in preprogrammed directions simultaneously.

2. Notation and Definitions

We use Σ to indicate an alphabet, a finite set of symbols. The cardinality of Σ is denoted with $\#\Sigma$. A finite sequence of symbols is called a word, or a one-dimensional block. The set of all words over alphabet Σ is denoted with Σ^* . The length of a word $w = a_1 \dots a_k$ is k and is denoted with $|w|$. The empty word is denoted with ε . The set of all words of length k is denoted with Σ^k , and set of all words of length less or equal to k is denoted with $\Sigma^{\leq k}$.

A *one-dimensional language* (further referred to as language) is a subset of Σ^* . For a language L , we extend the notation to $L_k = \Sigma^k \cap L$ and $L_{\leq k} = \Sigma^{\leq k} \cap L$. A word u such that $w = x_1 u x_2$ for some words $x_1, x_2 \in \Sigma^*$ is called a *factor of* w . The set of all factors of w is denoted with $F(w)$, and the set of all factors of length k of w is denoted with $F_k(w)$. We extend this notation such that the set of factors of all words in a language L denoted by $F(L)$ and the set of factors of all words in L of length k is denoted by $F_k(L)$.

Definition 2.1

A nondeterministic transducer is a five-tuple

$$\tau = (\Sigma, Q, \delta, q_0, T),$$

where Σ is a *finite alphabet*, Q is a *finite set of states*, q_0 is an *initial state* ($q_0 \in Q$), T is a *set of final (terminal) states* ($T \subseteq Q$), $\delta \subseteq Q \times \Sigma \times \Sigma \times Q$ is the set of transitions. A transducer is called deterministic if its set of transitions defines a function (which is also denoted with δ) $\delta: Q \times \Sigma \mapsto \Sigma \times Q$.

To a transducer we associate a directed labeled graph in the standard way: the set of vertices is the set of states Q and the directed edges are transitions in δ , such that an edge $e = (q, a, a', q')$ starts at q and terminates at q' . To each edge we associate labels $\mathfrak{I}: \delta \mapsto \Sigma$ and $\mathfrak{O}: \delta \mapsto \Sigma$ being the *input* and the *output* labels such that for $e = (q, a, a', q')$ we have $\mathfrak{I}(e) = a$ and $\mathfrak{O}(e) = a'$. The input and the output labels are naturally extended to paths in the transducer.

We say that a word w is *accepted* by a transducer $\tau = (\Sigma, Q, \delta, q_0, T)$ if there is a path $p = e_1 \dots e_k$ which starts at q_0 , terminates with a state in T and $\mathfrak{I}(p) = \mathfrak{I}(e_1) \dots \mathfrak{I}(e_k) = w$. The path p in this case is called an *accepting path for* w . The language which consists of all words that are accepted by τ is called the *input language* of τ and is denoted $\mathfrak{I}(\tau)$. A word v is said to be an *output* of τ if there is $w \in \mathfrak{I}(\tau)$ and an accepting path p for w such that $\mathfrak{O}(p) = v$. In this case we also write $v \in \mathfrak{O}(w)$. The language which consist of all outputs of τ is the *output language for* τ denoted with $\mathfrak{O}(\tau)$.

Definition 2.2

If $S = \{1, \dots, n\} \times \{1, \dots, m\}$, then a map $B: S \mapsto \Sigma$ is a *block* of size $n \times m$ or an $n \times m$ -*block*. We say that n is the *number of columns* in B and m is the *number of rows* in B . The empty block is of size $n \times m$ where at least one of m, n equals 0 and is denoted by ε .

The blocks are represented as arrays of symbols: the first row of the block is the bottom row, and every successive row is placed on top of the previous row.

Definition 2.3

If B is a block of size $n \times m$, and x, y, n', m' are positive integers, such that $x+n' = n+1$ and $y+m' = m+1$, then a sub-block of B of size $n' \times m'$ at position (x, y) , denoted $B|_{(x,y)}^{n' \times m'}$ is the $n' \times m'$ -block such that $B|_{(x,y)}^{n' \times m'}(i, j) = B(x+i-1, y+j-1)$ for $1 \leq i \leq n'$ and $1 \leq j \leq m'$. We say that a block B' is a sub-block of B if there are x, y, n', m' such that $B' = B|_{(x,y)}^{n' \times m'}$. We consider the empty block to be a sub-block of every block and we accept the convention that $B(1, 1)$ is the left bottom corner of B .

The set Σ^{**} denotes the set of all blocks over alphabet Σ . A subset L of Σ^{**} is called a *picture language*. The set of all $k \times \ell$ -blocks is denoted with $\Sigma^{k \times \ell}$. The set of all $k \times \ell$ sub-blocks of a block B is denoted by $F_{k, \ell}(B)$. The set of all sub-blocks of B is denoted with $F(B)$. We extend this notation naturally to $F(L)$ for all sub-blocks of blocks in L and $F_{k, \ell}(L)$ for the set of all sub-blocks of size $k \times \ell$ of blocks in L .

Now we concentrate on picture languages generated by transducers. If τ is a transducer (deterministic or nondeterministic), we define inductively: $\tau^0(w) = \{w\}$ if $w \in \mathcal{I}(\tau)$ and $\tau^i(w) = \{u \mid \text{there is } v \in \tau^{i-1}(w), u \in \tau(v)\}$.

Definition 2.4

An $n \times m$ -block B is *generated* by a transducer τ if $B|_{(1,1)}^{n \times 1} \in \mathcal{J}(\tau)$ and $B|_{(1,i+1)}^{n \times 1} \in \tau(B|_{(1,i)}^{n \times 1})$ for all $1 \leq i < m$.

Note that the requirement $B|_{(1,i+1)}^{n \times 1} \in \tau(B|_{(1,i)}^{n \times 1})$ is not equivalent to the more general property $B|_{(1,i+1)}^{1 \times m} \in \tau^i(B|_{(1,1)}^{1 \times m})$. Moreover the empty block ε is generated by a transducer τ if the initial state is also a terminal state. A $n \times 1$ -block is generated by a transducer τ if it belongs to $\mathcal{I}(\tau)$. Every transducer with non-empty input language generates blocks of size $n \times i$ for $i = 1, 2$.

Definition 2.5

A picture language which consists of all blocks that are generated by a deterministic (nondeterministic) transducer τ is called a picture language *generated by τ* and is denoted with L_τ . A picture language L is said to be *transducer generated* if there is a transducer τ such that $L = L_\tau$.

The class of transducer generated picture languages is denoted with TG.

3. Local Picture Languages and TG

Given an alphabet Γ and a one-dimensional language $L \subseteq \Gamma^*$ we say that L is *local* if there is $P \subseteq \Gamma^2$ such that $w \in L$ if and only if $F_2(w) \subseteq P$.

For a $n \times m$ -block B over Γ , let $\hat{B}$ denote the $(n+2) \times (m+2)$ -block obtained from B by surrounding B with symbols $\boxplus$. A two-dimensional language $L \subseteq \Gamma^{**}$ is *local* if there exists a finite set Θ of 2×2 -blocks over $\Gamma \cup \{\boxplus\}$ such that $L = \{B \in \Gamma^{**} \mid F_{2,2}(\hat{B}) \subseteq \Theta\}$ and we write $L = L(\Theta)$. We denote $\text{LOC}(\Gamma)$ (or simply LOC) the family of all local picture languages over Γ .

A two-dimensional language $L \subseteq \Gamma^{**}$ is *factorial local* (denoted FLOC) if there exists a finite set Θ of 2×2 blocks over Γ such that $L = F(L_\Theta)$ where $L_\Theta = \{B \in \Gamma^{**} \mid \emptyset \neq F_{2,2}(B) \subseteq \Theta\}$.

$\Theta\}$, and we write $L = L(\Theta)$. In FLOC languages the frames surrounding the blocks are removed since every sub-block of a block in the language is also in the language.

Proposition 3.1

For every $L \in \text{FLOC}$ there is a transducer τ such that $L_\tau = L$.

Proof—Let $L \in \text{FLOC}$. Then $L = L(\Theta)$ for some set of tiles Θ over alphabet Γ . Define a transducer $\tau = (\Gamma, Q, \delta, q_0, T)$ in the following way. The set of states Q is the set

$Q = \{q_0\} \cup \{B|_{(i,1)}^{1,2} \mid i=1, 2, B \in \Theta\}$ which besides the initial state contains all columns of blocks from Θ . The set of terminal states is $T = Q$. The transitions are the following:

- *start transitions:* (q_0, a, b, q) for all $q \in Q$ and $a \stackrel{b}{=} q$.
- *Θ transitions:* (q, a, b, q') whenever $qq' \in \Theta$ and $a \stackrel{b}{=} q'$.

Then $L = L_\tau$ follows directly from the fact that the input/output labels are always the same for all incoming edges to a state q and are equal to the bottom and the top symbols of q respectively. One transition follows another if and only if the end states of these transitions are the first and the second column of a block in Θ .

Hence we have shown that both languages contain the same blocks consisting of more than one row and one column. Since these languages consist precisely of the factors of such blocks, we conclude that $L = L_\tau$.

The following example shows that a transducer generated language may not be local even if the corresponding transducer is deterministic, and its input language is local.

Example 3.2

Consider the transducer over alphabet $\{a, b\}$ as depicted in Fig. 3.1. This transducer is deterministic and all of its states are terminal except the “junk state” q_3 . The initial state is q_0 . The output symbols for all transitions starting at the same state are the same. The input language of the transducer τ is local (the allowed words of length 2 are all but “bb”).

Suppose that L_τ is a local picture language and, there is a set Θ of allowed blocks of size $k \times k$ such that every $k \times k$ sub-block of a block in L_τ is in Θ . Let $B \times B'$ be the following $n \times m$ -blocks where $m, n > k$

$$B = \begin{array}{cccccc} a & a & a & \dots & a & a & a \\ & & \dots & & & & \\ a & a & a & \dots & a & a & a \\ a & b & a & \dots & a & a & b \end{array} \quad B' = \begin{array}{cccccc} a & a & a & \dots & a & a & a \\ & & \dots & & & & \\ a & a & a & \dots & a & a & b \\ b & a & a & \dots & a & b & a \end{array}.$$

$$C = \begin{array}{cccccc} a & a & a & \dots & a & a & a \\ & & \dots & & & & \\ a & a & a & \dots & a & a & b \\ a & b & a & \dots & a & b & a \end{array}.$$

And let C be an $(n+1) \times m$ -block:

One can think of C being a block obtained from B by adding the last column of B' as a last column. In this case, $B = C|_{(1,1)}^{n,m}$ and $B' = C|_{(2,1)}^{n,m}$, ie., $B, B' \in F(C)$. Quick check of the transducer shows that blocks B and B' are in L_τ . Moreover, $F_{k,k}(C) = F_{k,k}(\{B, B'\}) \subseteq \Theta$. If L_τ

is local, then it must be that $C \in L_{\tau}$, but direct check shows that it, in fact, is not (note that the only way to output symbol b is when the input word starts with b , and this is not the case with the first row of C).

A *tiling system* is a quadruple $(\Sigma, \Gamma, \Theta, \pi)$ where Σ and Γ are finite alphabets, Θ is a finite set of 2×2 blocks over $\Gamma \cup \{\Box\}$ defining a local language $L(\Theta)$ and $\pi: \Gamma \mapsto \Sigma$ is a map called *projection*. A two dimensional language $L \subseteq \Sigma^{**}$ is *tiling recognizable* if there exists a tiling system $(\Sigma, \Gamma, \Theta, \pi)$ such that $L = \pi(L(\Theta))$ (extending π to the blocks). We denote by $\text{REC}(\Sigma)$ (or simply REC) the family of all tiling recognizable two dimensional languages over Σ .

Proposition 3.3

Transducer generated languages are in REC .

Proof—Assume that the transducer $\tau = (\Sigma, Q, \delta, q_0, T)$ is given. Let $\Gamma = Q\Sigma = \{qa \mid q \in Q \text{ and } a \in \Sigma\}$ and projection $\pi: \Gamma \mapsto \Sigma$ be defined by $\pi(qa) = a$. To prove the claim we define a set of blocks Θ such that $\pi(L(\Theta)) = L_{\tau}$. The set Θ consists of the union of input blocks Θ_i , transition blocks Θ_t , and output blocks Θ_o defined below.

- *Input blocks.* The collection of input blocks Θ_i is the union of Θ_1, Θ_2 , and Θ_3 defined as follows.

$$\begin{aligned}\Theta_1 &= \left\{ \begin{array}{|c|c|} \hline qa & q'c \\ \hline \Box & \Box \\ \hline \end{array} \mid \exists b \in \Sigma, (q, a, b, q') \in \delta \text{ and } c \in \Sigma \right\} \text{ middle of the bottom row} \\ \Theta_2 &= \left\{ \begin{array}{|c|c|} \hline \Box & q_0a \\ \hline \Box & \Box \\ \hline \end{array} \mid q_0 \text{ is the initial state and } a \in \Sigma \right\} \text{ bottom left corner} \\ \Theta_3 &= \left\{ \begin{array}{|c|c|} \hline qa & \Box \\ \hline \Box & \Box \\ \hline \end{array} \mid \exists b \in \Sigma, \exists q' \in T, (q, a, b, q') \in \delta \right\} \text{ bottom right corner}\end{aligned}$$

- *Transition blocks.* The collection of transition blocks Θ_t is the union of Θ_4, Θ_5 , and Θ_6 defined as follows.

$$\begin{aligned}\Theta_4 &= \left\{ \begin{array}{|c|c|} \hline pb & p'd \\ \hline qa & q'd \\ \hline \end{array} \mid (q, a, b, q') \in \delta \text{ and } p, p' \in Q, c, d \in \Sigma \right\} \text{ middle } 2 \times 2 \text{-block simulating a transition in } \tau \\ \Theta_5 &= \left\{ \begin{array}{|c|c|} \hline \Box & q_0b \\ \hline \Box & q_0a \\ \hline \end{array} \mid \exists q \in Q, (q_0, a, b, q) \in \delta \right\} \text{ first left column} \\ \Theta_6 &= \left\{ \begin{array}{|c|c|} \hline pb & \Box \\ \hline qa & \Box \\ \hline \end{array} \mid p \in Q \text{ and } \exists q' \in T, (q, a, b, q') \in \delta \right\} \text{ the last column to the right}\end{aligned}$$

- *Output blocks.* The collection of output blocks Θ_o is the union of blocks Θ_7, Θ_8 , and Θ_9 defined below. Since the last row of a block generated by a transducer is not necessarily accepted by the transducer (i.e., it may not be in $\mathcal{J}(\tau)$) no transition type control is imposed on these tiles.

$$\begin{aligned}\Theta_7 &= \left\{ \begin{array}{|c|c|} \hline \Box & \Box \\ \hline qa & q'b \\ \hline \end{array} \mid qa, q'b \in \Gamma \right\} \\ \Theta_8 &= \left\{ \begin{array}{|c|c|} \hline \Box & \Box \\ \hline \Box & qa \\ \hline \end{array} \mid qa \in \Gamma \right\} \\ \Theta_9 &= \left\{ \begin{array}{|c|c|} \hline \Box & \Box \\ \hline qa & \Box \\ \hline \end{array} \mid qa \in \Gamma \right\}\end{aligned}$$

Assume that $B \in \pi(L(\Theta))$. Then $B = \pi(P)$ for some $n \times m$ -block $P \in L(\Theta)$. Let B_j and P_j denote the j -th rows of B and P respectively for $j = 1, \dots, m-1$. Then $B_j = j = a_1 \cdots a_n$ and there are states $q_1, \dots, q_{n-1}$ such that $P_j = q_0 a_1 \cdot q_1 a_2 \cdot \dots \cdot q_{n-1} a_n$. By construction of $L(\Theta)$ we have that $(q_{i-1}, a_i, b_i, q_i) \in \delta$ for some $b_i \in \Sigma$ for every $i = 0, \dots, n-1$, and the state q_n is in T . Hence the sequence $p_j = (q_0, a_1, b_1, q_1) \dots (q_{n-1}, a_n, b_n, q_n)$, is an accepting path in τ such that $B_j = \mathcal{J}(p_j) \in \mathcal{J}(\tau)$. Moreover, by construction of the transition blocks Θ_t we have that $B_{j+1} = b_1 \cdots b_n$ and so, $B_{j+1} = \mathcal{O}(p_j)$. Thus, $B \in L_\tau$ i.e., $\pi(L(\Theta)) \subseteq L_\tau$.

Conversely, if $B \in L_\tau$, then for each $1 \leq j < m-1$ there is an accepting path $p_j = (q_0, a_1, b_1, q_1)(q_1, a_2, b_2, q_2) \cdots (q_{n-1}, a_n, b_n, q_n)$ of τ such that $\pi(p_j) = B_j$ and $\mathcal{O}(p_j) = B_{j+1}$. Consider a $n \times m$ block P such that $P_j = q_0 a_1 \cdot q_1 a_2 \cdot \dots \cdot q_{n-1} a_n$ for all $1 \leq j < m-1$ and choose P_m to be a word over the alphabet Γ such that $\pi(P_m) = \mathcal{O}(p_{m-1}) = B_m$. Hence $\pi(P) = B$ and by construction $B \subseteq F_{2,2}(\Theta)$. Thus $B \in \pi(L(\Theta))$ and $\pi(L(\Theta)) = L_\tau$.

As noted with the following example there are picture languages in REC that are not in TG.

Example 3.4

Consider the picture language over alphabet $\{a, b\}$ which consists of all blocks containing at most one appearance of b (all other symbols are a). This picture language is in REC [19]. If there were a transducer that generates this language, this transducer would have a transition with an input symbol a and output b , and a transition with input symbol b and output a . But then, there are n, s such that $a^s b a^{n-s-1} \in \tau(a^n)$, i.e., on an input a^n the transducer may output a word containing a symbol b . Hence blocks with more than three rows that are generated by the transducer can have more than one appearance of b .

4. Entropy

One measure of complexity of a given language is the average growth rate of the number of blocks that appear in the language relative their size. In one dimensional case, this measure is also called the entropy of the language and is derived from the notion of topological entropy of (compact) symbolic dynamical systems (see [12]). The Perron-Frobenius theory provides a straight-forward way of computing the entropy for a one-dimensional regular language. It is the logarithm of the maximal eigen value (also known as Perron numbers) of the adjacency matrix of the minimal deterministic automaton of the language [25]. However, obtaining a general method for computing the entropy of a two-dimensional recognizable language shows to be very difficult, and consequently, there is no general knowledge about what numbers can be realized as the entropy values in two-dimensions. In the following we show that logarithm of every Perron number can be obtained as an entropy value for a transducer generated picture language. In other words, all entropy values realized by one-dimensional regular languages can also be realized by transducer generated picture languages.

Recall that for $L \subseteq \Sigma^*$ the *entropy of L* is $h(L) = \limsup_{n \rightarrow \infty} \frac{1}{n} \log(\#F_n(L))$. This definition extends to an equivalent definition in two-dimensions.

Definition 4.1

For $L \in \Sigma^{**}$ the entropy of the picture language L is

$$h(L) = \limsup_{n \rightarrow \infty} \frac{1}{n^2} \log(\#F_{n,n}(L)).$$

Note

- a. Since the entropy formula “counts” the sub-blocks of a certain size, the value of the entropy of L_τ and $\mathcal{J}(\tau)$ does not change if we consider these languages to be factorial, hence we can assume that all transducers in this sections have all their states initial and terminal. In this case, $F(L_\tau) = L_\tau$.
- b. It can be shown that in this case $\limsup$ is always finite [12], and moreover, that the limit always exists. Hence we write $\lim$ instead of $\limsup$.

Note that the entropy of L is always bounded by $\log \#\Sigma$. The entropy of L_τ is zero for any deterministic transducer τ , since for each n it contains at most $\#\Sigma^n$ blocks of size $n \times n$ (each word w can generate at most one block of height n). However, if L_τ is a two dimensional language generated by a nondeterministic transducer, its entropy may no longer be zero. For example, the transducer $\tau = (\Sigma, \{q\}, \delta, q, \{q\})$ with $\delta = \{(q, a, b, q) | a, b \in \Sigma\}$, which consists of one state with transitions having all possible labels generates $L_\tau = \Sigma^{**}$ and for each n , $\#F_{n,n}(L_\tau) = \#(\Sigma^n \times \Sigma^n) = \#\Sigma^{2n}$. Thus $h(L_\tau) = h(\Sigma^{**}) = \log \#\Sigma$.

For a given transducer τ and w in $\mathcal{J}(\tau)$ define $\deg(w) = \#\tau(w)$, i.e., $\deg(w)$ is the number of distinct words that τ can output on input w . We extend this notation to finite sets of words by setting $\deg(S) = \max_{w \in S} \{\deg(w)\}$.

Observe that the number of distinct blocks of height n with the bottom row w contained in L_τ is bounded above by

$$\prod_{i=0}^{n-1} \deg(\tau^i(w)).$$

Suppose there is $k > 0$ such that $\deg(w) \leq n^k$ for all w in $\mathcal{J}(\tau)$ with $|w| = n$. Then there are at most $(n^k)^n = n^{nk}$ $n \times n$ -blocks. We observe that

$$\begin{aligned} h(L_\tau) &= \lim_{n \rightarrow \infty} \frac{1}{n^2} \log(\#F_{n,n}(L_\tau)) \\ &\leq \lim_{n \rightarrow \infty} \frac{1}{n^2} \log \left(\sum_{\substack{|w|=n \\ w \in \mathcal{J}(\tau)}} \left(\prod_{i=0}^{n-1} (\deg(\tau^i(w))) \right) \right) \\ &\leq \lim_{n \rightarrow \infty} \frac{1}{n^2} \log(\#\Sigma^n \cdot n^{nk}) = \lim_{n \rightarrow \infty} \left[\frac{1}{n} \log \#\Sigma + \frac{k}{n} \log n \right] = 0. \end{aligned}$$

Thus if there is some positive integer k , such that for any w in $\mathcal{J}(\tau)$, $\#\tau(w) \leq |w|^k$, the entropy of L_τ is zero. This proves:

Proposition 4.2

Let τ be a transducer. If there is a polynomial p such that $\#\tau(w) \leq p(|w|)$ for every $w \in \mathcal{J}(\tau)$, then the entropy of L_τ is 0.

Proposition 4.3

For any regular language L there is a nondeterministic transducer τ such that $\mathcal{J}(\tau) = L$ and $h(L_\tau) = h(L)$.

Proof—Let M be a finite state automaton accepting language L . Denote the set of transitions of M by φ , and construct a transducer τ from M as follows. The set of states in τ remains the same as in M and the set of transitions in τ is $\{(q_1, a, b, q_2) \mid (q_1, a, q_2) \in \varphi \text{ and } b \in \Sigma\}$. Observe that $\mathcal{J}(\tau) = L$.

Now consider L_τ . Note that for an $n \times n$ -block $B \in L_\tau$, every row of B must be in $\mathcal{J}(\tau)$, except maybe the last, n^{th} row which may not belong to $\mathcal{J}(\tau)$. Let $B_{p,k}$ denote the set of all $n \times p$ -blocks $B \in L_\tau$ with $B|_{(1,p)}^{n,1}$ (the p^{th} row of block B) in $\mathcal{J}(\tau)$. Thus we conclude that $\#F_{n,n}(L_\tau) \leq \#B_{n,n-1}\#\Sigma^n \leq \#B_{n,n}\#\Sigma^n$. Hence

$$\begin{aligned} h(L_\tau) &= \lim_{n \rightarrow \infty} \frac{1}{n^2} \log(\#F_{n,n}(L_\tau)) \\ &\leq \lim_{n \rightarrow \infty} \frac{1}{n^2} \log(\#B_{n,n} \cdot \#\Sigma^n) \\ &= \lim_{n \rightarrow \infty} \frac{1}{n^2} \log \#B_{n,n} \end{aligned}$$

Since $B_{n,n} \subseteq F_{n,n}(L_\tau)$, we have $\#B_{n,n} \leq \#F_{n,n}(L_\tau)$, hence

$$h(L_\tau) = \lim_{n \rightarrow \infty} \frac{1}{n^2} \log \#B_{n,n}$$

Because the transducer can output every word of length n on any given input of length n , the number of $n \times n$ -blocks in L_τ having each of its rows in $\mathcal{J}(\tau) = L$ is exactly $[\#F_n(\mathcal{J}(\tau))]^n = (F_n(L))^n$, i.e., $\#B_{n,n} = [\#F_n(L)]^n$. Thus

$$h(L_\tau) = \lim_{n \rightarrow \infty} \frac{1}{n} \log \#F_n(L) = h(L)$$

Corollary 4.4

Let L be a regular language. Then the entropy of L is an upper bound for the entropy of L_τ for any nondeterministic transducer τ with $\mathcal{J}(\tau) = L$.

Proof—Let L be a regular language and let $\tau = (A, Q, \delta, q_0, F)$ be such that $\mathcal{J}(\tau) = L$. Then consider $\tau' = (A, Q, \delta', q_0, F)$, with $\delta' = \{(q, a, b, p) \mid (q, a, s, p) \in \delta \text{ and } b \in \Sigma\}$. Hence $\delta \subseteq \delta'$ and $h(L_\tau) = h(L_{\tau'})$. By the proof of the Proposition 4.3, $h(L_{\tau'}) = h(\mathcal{J}(\tau)) = h(L)$.

5. Representing Transducers with Wang Tiles

The relationship between transducers and Wang tiles has been used by J. Kari [16] for generating small set of aperiodic Wang tiles, and the idea that iteration of transducers can correspond to some two-dimensional local languages was implicitly used in [23]. In [13] it was observed that composition of transducers can generate all computable functions. The transducer generated languages are not necessarily local, as they do not necessarily correspond to the languages associated with Wang tiles (e.g., Example 3.2). However, there is a natural way to describe the transducer transitions by Wang tiles as shown below. We use the projection of these Wang tiles on their bottom color to represent the strings outputted by the transducer and to guide the arrangements of the robotic arms in arrays.

A finite set of distinct unit squares with colored edges is called a set of *Wang prototiles*. We assume that each prototile has an arbitrarily large number of copies called *tiles*. A tile ξ with left edge colored l , bottom edge colored b , top edge colored t and right edge colored r is denoted with $\tau = [l, b, t, r]$. No rotation or reflexion of the tiles is allowed. Two tiles $\xi = [l, b, t, r]$ and $\xi' = [l', b', t', r']$ can be placed next to each other, ξ to the left of ξ' iff $r = l'$, and ξ' on top of ξ iff $t = b'$.

To each transducer τ we associate a collection of Wang tiles.

- *Input and output tiles.* Four fixed colors, denoted $\beta_l, \beta_b, \beta_r, \beta_t$, called *borders*, are used in the set of colors, each representing one of the left, bottom, right and top border. For each symbol $a \in A$ an *input* prototile $\xi_a = [c, \beta_b, a, c]$ is added. There are $\# \Sigma$ such input prototiles. The left and right sides are colored with a new color which is the same for every symbol a , called the *connect* color c . The “start of input” prototile $[\beta_l, \beta_b, c_{q_0}, c]$ (the color c_{q_0} indicates the start of the computation) and the “end of input” prototile $[c, \beta_b, a, \beta_r]$ (a is an end of input symbol, allowing to add the right border color) are also added. The output prototiles which “cap-off” the computation are essentially the same as the input tiles, except they have the “top” border: $\xi'_a = [c', a, \beta_t, c']$ where c' is another connect color. The output row of tiles starts with a top-left corner tile $[\beta_b, c_{q_0}, \beta_b, c']$ and ends with a top-right corner $[c', a, \beta_b, \beta_r]$.

We denote this set with B_τ .

- *Computational tiles.* Given transducer τ , to each transition of the form (q, a, a', q') we associate a prototile $[q, a, a', q']$ as presented in Fig. 5.1. Hence, there are as many computational prototiles as there are transitions in τ . Additional left border tile $[\beta_b, c_{q_0}, c_{q_0}, q_0]$ is added in order to start the computation of the transducer. Similarly, right border tiles $[q, a, a, \beta_r]$ are added for each terminal state q .

We call this set of prototiles C_τ .

- *Finite state automaton tiles.* Single row blocks in the language L_τ are precisely the blocks that define $\mathcal{J}(\tau)$. In order to generate these blocks with Wang tiles, we include prototiles that simulate the transitions of the transducer as a finite state automaton, but instead of output, the top color is immediately the border color. Hence, to start reading the input, we include the prototile $[\beta_b, c_{q_0}, \beta_b, q_0]$. For each transition of the form (q, a, a', q') in τ we include a prototile $[q, a, \beta_b, q']$, and finally we include prototiles $[q, a, \beta_b, \beta_r]$ for each terminal state $q \in T$.

We denote this set of prototiles A_τ .

The Wang prototiles that correspond to the transducer τ are $\mathcal{T}_\tau = A_\tau \cup B_\tau \cup C_\tau$. With these sets of input and output prototiles, every iterated computation of τ can be modeled as a tiled rectangle surrounded by boundary colors. In particular one run of the transducer τ reading an input of length n and producing an output is obtained with a tiled $n \times 3$ rectangle ($n > 2$) such that the sides of the rectangle are colored with boundary colors. As an example consider the transducer T presented in Fig. 5.2(a). It has initial and terminal state s_0 with input/output alphabet $\{0, 1\}$. It accepts the set of binary strings that represent numbers divisible by 3. The output is the result of the division of the binary string with three. On input 10101 (21 in decimal) the transducer gives the output 00111 (7 in decimal). The states s_0, s_1, s_2 represent the remainder of the division of the input string with 3. Instead of the last row with boundary color on top, an additional row of tiles can stack on top of the first computation, iterating the transducer by taking its output as a new input.

The set of all rectangular tilings with border colors on top, left, bottom and right obtained from tiles in $\mathcal{T}_\tau$ is denoted with $W(\mathcal{T}_\tau)$. Consider the projection $\pi: \mathcal{T}_\tau \rightarrow \Sigma$ projecting on the bottom color of a tile defined with $\pi([l, b, t, r]) = b$. Then, directly from the construction we have:

Proposition 5.1

A rectangular $(n + 2) \times (m + 1)$ -array of tiles P is in $W(\mathcal{T}_\tau)$ if and only if there is an $n \times m$ -block B in L_τ such that $B(i, j) = \pi(P(i + 1, j + 1))$ for all $i = 1, \dots, n$ and $j = 1, \dots, m$.

The above proposition indicates that blocks in L_τ are obtained from rectangular arrays in $W(\mathcal{T}_\tau)$ by removing boundary tiles (left and right sides, and the bottom boundary tiles) and projecting the remaining tiles to their bottom color. For example, the rectangular 7×3 -array

$$\begin{array}{ccccc} & & 0 & 0 & 1 & 1 & 1 \\ & & 0 & 0 & 1 & 1 & 1 \end{array}$$

of tiles in Figure 5.2(right) defines the 5×2 -block $\begin{array}{ccccc} 1 & 0 & 1 & 0 & 1 \end{array}$.

6. Molecular Implementation

The base row of the two-dimensional array of molecular robotic arms arrangement can be obtained through a set of TX molecules that simulates the execution of a transducer.

Wang tiles are simulated by DNA TX molecules. The colors represent “sticky ends” in the TX molecule and the border colors represent “hairpin ends” such that no assembly is allowed after the border. Each prototile may appear in large number of molecular copies as separately annealed DNA TX molecules. No rotation of the tiles is assured by the coding of the sticky ends. The basic TX tile representing one transition of the transducer schematically is depicted in Fig. 6.1.

The transducer transitions use tiles made of triple crossover (TX) DNA molecules with sticky ends corresponding to one side of the Wang tile. An example of such molecule is presented in Fig. 6.1(right) with 3' ends indicated with arrowheads. The connector is a sticky part that does not contain any information but it is necessary for the correct assembly of the tiles. These TX tiles have the middle duplex longer than the other two. It has been shown that such TX molecules can self-assemble in an array [22]. In [5], an implementation of the transducer represented in Figure 5.2 on variable, programmable inputs is reported. The input is adjusted through a sequence of PX-JX₂ devices. In [31] the sequence-dependent PX-JX₂ 2-state nanomechanical device is reported. This robust device, whose machine cycle is shown in Fig. 6.2 is directed by the addition of set strands to the solution that forms its environment. The set strands, drawn green and purple, establish which of the two states the device assumes. They differ by rotation of a half-turn. The sequence-driven nature of the device means that many different devices can be constructed, each of which is individually addressable; this is done by changing the sequences of the strands connecting the DX portion AB with the DX portion A'B' where the green or purple strands pair with them. The green and purple strands have short, unpaired extensions on them. The state of the device is changed by binding the full biotin-tailed complements of green or purple strands, removing them from solution by magnetic streptavidin beads, and then adding the other strands. In [24, 9] it was shown that these devices can be linearly ordered and each can be individually addressable. Precisely this property of the device is used to obtain a programmable input for the transducer in [5]. Linear arrays of a series of PX-JX₂ devices have been adapted to set the input of the transducer. The computational set-up corresponding to the Wang tiles in Fig. 5.2(b) is illustrated schematically in Fig. 6.3. In Fig. 6.3 one base row corresponding to the input is depicted, as in the example shown in Fig. 5.2 with computational tiles made of DNA triple crossover (TX) molecules.

6.1. Arranging Devices in an Array

A series of PX-JX₂ devices inserted into gaps in a 2D TX array is reported in [6]. Each of these devices is controlled individually, corresponding to the presence of a particular strand in the solution (n devices lead to 2^n states). The developed cassette is shown schematically in both of its states in Fig. 6.4. Portions (a) and (b) show the cassette in the PX state; (a) is perpendicular to its plane, and (b) is oblique (parts (c) and (d) show similar views of the JX₂ state). The cassette consists of three helical domains, one of which is much shorter than the other two. The short domain is the insertion domain. It contains sticky ends that enable its cohesion roughly perpendicular ($\sim 103^\circ$) to the array that supports it [6]. The other two domains are the device in either the PX or JX₂ state. The array that was selected for insertion is the triple crossover (TX) tile array [22]. In this array, the TX tiles are connected as shown in Fig. 6.5.

6.2. Arranging Robotic Arms in a Programmed Array

A base of the array is programmed such that the PX-JX₂ devices can be placed in a predefined arrangement. The base row generated by one execution of the transducer defines the first arrangement of the devices. This is depicted in Fig. 6.6. The linear assembly of the bottom devices (input devices) is the arrangement which defines an input. With different choices of positions of the devices in PX or JX₂ positions, the input symbol is specified as a or b . The chosen input in the figure is $ababa$ which corresponds to the binary string 10101. The output produced by the transducer is the string $bbaaa$ representing the binary string 00111.

The output sets the choice of particular cassettes (see Fig. 6.4) to be placed in the array. Cassettes containing the device in a JX₂ form are encoded to correspond to output b and a different cassette (with opposite control sequences and different sticky ends in its binding domain) containing the device in a PX form are encoded to correspond to output a . Hence, the same set of set strands shall move the devices from JX₂ into PX and the devices from PX into JX₂ form. This is done using control sequences on the device frames that put one device in the PX state, and the other device in the JX₂ state, using the same set strands; the other set strands achieve exactly the opposite result. The continuation of the assembly into a larger array is similar to the one shown in Fig. 6.5 except, now the TX molecules within the array would transfer the code of the output and all subsequent rows with cassettes would contain the same PX and JX₂ initial setting of the devices. This is represented schematically by the light and dark blue TX tiles in Fig. 6.6. By applying the set strands that exchange the devices from one state into another the arms (hairpins) can be moved synchronously from one position into another (schematically, the arms being “up” change to “down”, and those being “down” change to “up”).

A transducer generated array can be obtained by allowing third, fourth etc. rows of assembly. Note that it is not necessary that transition molecules be implemented immediately on the input. Hence, the first arrangement of the devices can be specified by the defined input, but after the first block of the arranged devices, the transition molecules can be applied and the next assembly of the array could follow the arrangement of the devices as a result of the transducer implementation. Of course, one can apply the transition molecules after the second block of arrangements and the subsequent arrangement of the devices would be according to the next output. This is schematically presented in Fig. 6.7 where it is shown the change of the positions of the arms after the set strands have been applied. The arrangement of the arms in Fig. 6.7 corresponds to the 5×2 -block obtained as a projection of the Wang tile array in Fig. 5.2(right). Two symbols of the input a and b are represented with squares colored light and dark blue respectively. The TX transition molecules are represented with squares colored yellow. Cassettes and the arms made of hairpins are

colored red. On the left of the figure, the position of the arms is according to the initial state of the devices, and on the right it is in a position after the set strands have been applied resulting in a synchronous movement of the arms.

7. Concluding Remarks

This paper introduces a class of two-dimensional arrays of symbols generated by transducers. This class consists of languages which are included in REC and also contain all factorial local picture languages. If entropy is taken as a measure of complexity, this class shows to have rich pattern generation capabilities. Characterization of patterns that can be generated by iteration of transducers may be of interest in applications, in particular, in algorithmic self-assembly of two-dimensional arrays with DNA tiles. Also, it may be of interest to investigate some decidability questions for these languages: for example, given a language in TG, is there an $m \times n$ -block in the language for every m, n ? The relationship of the class TG with the class of unambiguous (or non-deterministic) picture languages as well as transitivity and mixing properties of these languages remain to be investigated as well.

Acknowledgments

This work has been supported in part by the NSF grants CCF #0523928 and CCF #0726396 to N.J. and by grants GM-29554 from NIGMS, grants DMI-0210844, EIA-0086015, CCF-0432009, CCF-0523290 and CTS-0548774, CTS-0608889 from the NSF, 48681-EL and W911NF-07-1-0439 from ARO, N000140910181 from the office of Naval Research and a grant from the W.M. Keck Foundation, to N.C.S.

References

1. Anselmo, M.; Giammarresi, D.; Madonia, M. Tiling automaton: a computational model for recognizable two-dimensional languages. In: Holub, J.; Žd'árek, J., editors. CIAA 2007. Vol. 4783. Springer-Verlag; Berlin Heidelberg: 2007. p. 290-302.LNCS
2. Anselmo, M.; Giammarresi, D.; Madonia, M. From determinism to non-determinism in recognizable two-dimensional languages. In: Harju, T.; Karhumäki, J.; Lepistö, A., editors. DLT 2007. Vol. 4588. Springer-Verlag; Berlin Heidelberg: 2007. p. 36-47.LNCS
3. Anselmo M, Giammarresi D, Madonia M, Restivo A. Unambiguous recognizable two-dimensional languages. *RAIRO-Inf Theor Appl*. 2006; 40:277–293.
4. Anselmo, M.; Jonoska, N.; Madonia, M. Framed versus Unframed Two Dimensional Languages. In: Nielsen, M., et al., editors. *SOFSEM 09*. Vol. 5404. 2009. p. 79-92.LNCS
5. Chakraborty, B.; Jonoska, N.; Seeman, NC. Programmable transducer by DNA self-assembly. submitted
6. Ding B, Seeman NC. Operation of a DNA Robot Arm Inserted Into a 2D DNA Crystalline Substrate. *Science*. 2006; 314:1583–1585. [PubMed: 17158323]
7. Dolzhenko, E.; Jonoska, N. On complexity of two dimensional languages generated by transducers. In: Ibarra, O., et al., editors. *Implementation and Application of Automata*. Vol. 5148. Springer; 2008. p. 181-190.LNCS
8. Fu TJ, Seeman NC. DNA double crossover structures. *Biochemistry*. 1993; 32:3211–3220. [PubMed: 8461289]
9. Garibotti AV, Liao S, Seeman NC. A Simple DNA-Based Translation System. *NanoLetters*. 2007;7. in press.
10. Giammarresi D, Restivo A, Nivat M, Saoudi A, Wang PSP. Recognizable picture languages. *Proc. 1st Internat. Colloq. on Parallel Image Processing* also in *Internat J Pattern Recognition Artif Intell*. 1992; 6:231–256.
11. Giammarresi, D.; Restivo, A. *Handbook of Formal Languages*. Vol. 3. Springer; Berlin: 1997. Two-dimensional languages; p. 215-267.
12. Gottschalk, WH.; Hedlund, GA. *Topological Dynamics*. Vol. 36. AMS Colloquium Pubs; 1955.

13. Jonoska, N.; Liao, S.; Seeman, NC. Transducers with programmable input by DNA self-assembly. In: Jonoska, N.; Paun, Gh; Rozenberg, G., editors. Aspects in Molecular Computing. Vol. 2950. Springer; 2004. p. 219-240. Lecture Notes in Computer Science
14. Jonoska N, Pirnot JB. Transitivity in two-dimensional local languages defined by dot systems. International Journal of Foundations of Computer Science. 2006; 17:435–464.
15. Jonoska, N.; Pirnot, JB. Finite state automata representing two-dimensional subshifts. In: Holub, J.; Žd'árek, J., editors. CIAA 2007. Vol. 4783. 2007. p. 277-289. LNCS
16. Kari J. A small aperiodic set of Wang tiles. Discrete Math. Nov; 1996 160(1–3):259–264.
17. Kari, J.; Moore, C. Rectangles and squares recognized by two-dimensional automata. <http://www.santafe.edu/~moore/pubs/picture.html>
18. Kari J, Moore C. New results on alternating and non-deterministic two-dimensional finite automata in. Proceedings of STACS 2001, LNCS. 2001; 2010:396–406.
19. Kari, J. private communication.
20. Kitchens B, Schmidt K. Automorphisms of compact groups. Ergod Th & Dynam Sys. 1989; 9:691–735.
21. Kitchens, B.; Schmidt, K. Markov subgroups of $(\mathbb{Z}/2\mathbb{Z})^{\mathbb{Z}^2}$. In: Walters, P., editor. Symbolic Dynamics and its Applications. Vol. 135. 1992. p. 265-283. Contemporary Mathematics
22. LaBean TH, Yan H, Kopatsch J, Liu F, Winfree E, Reif JH, Seeman NC. J Am Chem Soc. 2000; 122:1848–1860.
23. Latteux, M.; Simplot, D.; Terlutte, A. Iterated Length-Preserving Rational Transductions. In: Brim, L., et al., editors. MFCS '98. Vol. 1450. 1998. p. 286-295. LNCS
24. Liao S, Seeman NC. Translation of DNA Signals into Polymer Assembly Instructions. Science. 2004; 306:2072–2074. [PubMed: 15604403]
25. Lind, D.; Marcus, B. An Introduction to Symbolic Dynamics and Coding. Cambridge University Press; Cambridge: 1995.
26. Manca V, Martin-Vide C, Paun Gh. New computing paradigms suggested by DNA computing: computing by carving. BioSystems. 1999; 52:47–54. [PubMed: 10636029]
27. Paun, Gh. On the iteration of gsm mappings. Rev Roum Math Pures Appl. 1978; 234:921–937.
28. Rothmund P, Papadakis N, Winfree E. Algorithmic Self-assembly of DNA Sierpinski Triangles. PLoS Biology. 2004; 2(12):e424. <http://biology.plosjournals.org/perlserv/?request=get-document&doi=10.1371/journal.pbio.0020424&ct=1>. [PubMed: 15583715]
29. Winfree E, Liu F, Wenzler LA, Seeman NC. Design and self-assembly of two-dimensional DNA crystals. Nature. 1998; 394:539–544. [PubMed: 9707114]
30. Winfree E. Algorithmic self-assembly of DNA: theoretical motivations and 2D assembly experiments. Journal of Biomolecular Structure and Dynamics. 2000; 11(S2):263–270. [PubMed: 22607433]
31. Yan H, Zhang X, Shen Z, Seeman NC. A robust DNA mechanical device controlled by hybridization topology. Nature. 2002; 415:62–65. [PubMed: 11780115]

Figure 3.1.

A deterministic transducer τ such that $\mathcal{J}(\tau)$ is a local language, but L_τ is not.

Figure 5.1.
A computational tile for a transducer.

Figure 5.2.

Left: a transducer that outputs the result of dividing the binary string with 3 in binary; right: tiles representing a single computation of the transducer reading an input 10101 and producing output 00111. The shades of brown indicate the border colors, and shades of blue indicate the connectors c and c' . The gold to the left represents c_{q0} and the pink to the right is the end of input symbol α . The states colors are shades of purple whereas the symbol 0 is red and the symbol 1 is white.

Figure 6.1.
A basic TX tile representing one transition of the transducer.

Figure 6.2.
The PX-JX₂ device.

Figure 6.3.
Schematic view of the base-row programming the array.

Figure 6.4. Schematic view of the cassette containing the devices. View parallel to the array (a) and perpendicular from the top of the array (b).

Figure 6.5. Schematic view of the TX array with two distinct positions of the arms obtained through the two distinct states of the devices.

Figure 6.6.
Schematic view of the programmed device arrangement

Figure 6.7.
Simultaneous movement of the arms in a programmed arrangements of devices.