Tile Rewriting Grammars and Picture Languages [1]

Stefano Crespi Reghizzi, Matteo Pradella

DEI - Politecnico di Milano and
CNR IEIIT-MI,
Piazza Leonardo da Vinci, 32,
I-20133 Milano, Italy
e-mail: {crespi, pradella}@elet.polimi.it

Abstract

Tile Rewriting Grammars (TRG) are a new model for defining picture languages. A rewriting rule changes a homogeneous rectangular subpicture into a isometric one tiled with specified tiles. Derivation and language generation with TRG rules are similar to context-free grammars. A normal form and some closure properties are presented. We prove this model has greater generative capacity than the Tiling Systems of Giammarresi and Restivo and the grammars of Matz, another generalization of context free string grammars to 2D. Examples are shown for pictures made by nested frames and spirals.

Key words: picture languages, 2D languages, tiling systems, context-free grammars, locally testable languages.

1 Introduction

In the past several proposals have been made for applying to pictures (or 2D) languages the generative grammar approach but in our opinion none of them matches the elegance and descriptive adequacy that made Context Free (CF) grammars so successful for string languages. A picture is a rectangular array of terminal symbols (the pixels).

A survey of formal models for picture languages is [3] where different approaches are compared and related: tiling systems, cellular automata, and grammars. The lat-

[1] A preliminary version is [6]. Work partially supported by MIUR, Progetto *Linguaggi formali e automi, teoria e applicazioni*.

ter had been surveyed in more detail by [7]. Classical 2D grammars can be grouped in two categories [2] called *matrix* and *array* grammars respectively.

The matrix grammars, introduced by A. Rosenfeld, impose the constraint that the left and right parts of a rewriting rule must be isometric arrays; this condition overcomes the inherent problem of "shearing" which pops up while substituting a subarray in a host array.

Siromoney's array grammars are parallel-sequential in nature, in the sense that first a horizontal string of nonterminals is derived sequentially, using the horizontal productions; and then the vertical derivations proceed in parallel, applying a set of vertical productions. Several variations have been made, for instance [1]. A particular case are the 2D right-linear grammars in [3].

Matz's *context-free picture grammars* [5] rely on the notion of row and column concatenation and their closures. A rule is like a string CF one, but the right part is a 2D regular expression. The shearing problem is avoided because, say, row concatenation is a partial operation which is only defined on pictures of identical width.

Exploring a different course, our new model, *Tile Rewriting Grammar* (TRG), intuitively combines Rosenfeld's isometric rewriting rules with the Tiling System (TS) of Giammarresi and Restivo [2]. The latter defines the family of Recognizable 2D languages (the same accepted by *on-line tessellation automata* of Inoue and Nakamura [4]).

A TRG rule is a schema having to the left a nonterminal symbol and to the right a *local* 2D language over terminals and nonterminals; that is the right part is specified by a set of fixed size tiles.

As in matrix grammars, the shearing problem is avoided by a isometric constraint, but the size of a TRG rule needs not to be fixed. The left part denotes any rectangle filled with the same nonterminal. Whatever size the left part takes, the same size is assigned to the right part. To make this idea effective, we impose a tree partial order on the areas which are rewritten. A progressively refined equivalence relation implements the partial ordering. Derivations can then be visualized in 3D as well nested prisms, the analogue of syntax trees of string grammars.

To our knowledge this approach is novel and is able to generate an interesting gamut of pictures: grids, spirals, and in particular a language of nested frames, which is in some way the analogue of a Dyck language.

Sect. 2 lists the basic definitions. Sect. 3 presents the definition of TRG grammar and derivation, two examples, and proves the basic properties of the model: canonical derivation, uselessness of concave rules, normal forms, closures for some opera-

[2] Leaving aside the graph grammar models because they generate graphs, not 2D matrices.

tions. Sect. 3 compares TRG with other models, proving that its generative capacity exceeds that of Tiling Systems and of Matz's CF picture grammars. The appendix contains the grammar of Archimedes spirals.

2 Basic Definitions

Many of the following notation and definitions are from [3].

Definition 1 *For a finite alphabet Σ, the set of pictures is Σ^{**}. For $h, k \geq 1$, $\Sigma^{(h,k)}$ denotes the set of pictures of size (h,k) (we will use the notation $|p| = (h,k), |p|_{row} = h, |p|_{col} = k$). # is used when needed as a* boundary symbol*; $\hat{p}$ refers to the bordered version of picture p. That is:*

$$p \in \Sigma^{(h,k)} \equiv p = \begin{matrix} p(1,1) & \dots & p(1,k) \\ \vdots & \ddots & \vdots \\ p(h,1) & \dots & p(h,k) \end{matrix} \qquad \hat{p} = \begin{matrix} \# & \# & \dots & \# & \# \\ \# & p(1,1) & \dots & p(1,k) & \# \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \# & p(h,1) & \dots & p(h,k) & \# \\ \# & \# & \dots & \# & \# \end{matrix}$$

A pixel *is an element $p(i,j)$. If all pixels are identical to $C \in \Sigma$ the picture is called* homogeneous *and denoted as C-picture.*

Row and column concatenations *are denoted $\ominus$ and $\oplus$, respectively. $p \ominus q$ is defined iff p and q have the same number of columns; the resulting picture is the vertical juxtaposition of p over q. $p^{k\ominus}$ is the vertical juxtaposition of k copies of p; $p^{*\ominus}$ is the corresponding closure. $\oplus, ^{k\oplus}, ^{*\oplus}$ are the column analogous.*

The pixel-by-pixel cartesian product *(written $p \otimes q$) is defined iff $|p| = |q|$ and is such that for all i, j, $(p \otimes q)(i,j) = \langle p(i,j), q(i,j) \rangle$.*

Definition 2 *Let p be a picture of size (h,k). A* subpicture *of p at position (i,j) is a picture q such that, if (h',k') is the size of q, then $h' \leq h, k' \leq k$, and there exist integers i, j $(i \leq h-h'+1, j \leq k-k'+1)$ such that $q(i',j') = p(i+i'-1, j+j'-1)$ for all $1 \leq i' \leq h', 1 \leq j' \leq k'$. We will write also $q \trianglelefteq_{(i,j)} p$, or the shortcut $q \trianglelefteq p \equiv \exists i,j (q \trianglelefteq_{(i,j)} p)$.*

Moreover, if $q \trianglelefteq_{(i,j)} p$, we define $coor_{(i,j)}(q,p)$ as the set of coordinates of p where q is located:

$$coor_{(i,j)}(q,p) = \{(x,y) \mid i \leq x \leq i + |q|_{row} - 1 \wedge j \leq y \leq j + |q|_{col} - 1\}$$

Conventionally, $coor_{(i,j)}(q,p) = \emptyset$, if q is not a subpicture of p. If q coincides with p we write $coor(p)$ instead of $coor_{(1,1)}(p,p)$.

Definition 3 *Let γ be an equivalence relation on $coor(p)$, written $(x,y) \overset{\gamma}{\sim} (x',y')$. Two subpictures $q \trianglelefteq_{(i,j)} p, q' \trianglelefteq_{(i',j')} p$ are* γ-equivalent, *written $q \overset{\gamma}{\sim} q'$, iff for all pairs $(x,y) \in coor_{(i,j)}(q,p)$ and $(x',y') \in coor_{(i',j')}(q',p)$ it holds $(x,y) \overset{\gamma}{\sim} (x',y')$.*

A homogeneous C-subpicture $q \trianglelefteq p$ is called maximal with respect to relation γ *iff for every γ-equivalent C-subpicture q' it is*

$$coor(q,p) \cap coor(q',p) = \emptyset \vee coor(q',p) \subseteq coor(q,p).$$

In other words q is maximal if any C-subpicture which is equivalent to q is either a subpicture of q or it is not overlapping. [3]

Definition 4 *For a picture $p \in \Sigma^{**}$ the* set of subpictures *(or* tiles*) with size (h,k) is:*

$$B_{h,k}(p) = \{q \in \Sigma^{(h,k)} \mid q \trianglelefteq p\}.$$

We assume $B_{1,k}$ to be only defined on $\Sigma^{(1,*)}$ (horizontal strings), and $B_{h,1}$ on $\Sigma^{(*,1)}$ (vertical strings).

For brevity, for tiles of size $(1,2)$, $(2,1)$, or $(2,2)$, we introduce the following notation:

$$[\![p]\!] = \begin{cases} B_{1,2}(p), \text{if } |p| = (1,k), k > 1 \\ B_{2,1}(p), \text{if } |p| = (h,1), h > 1 \\ B_{2,2}(p), \text{if } |p| = (h,k), h,k > 1 \end{cases}$$

Definition 5 *Consider a set of tiles $\omega \subseteq \Sigma^{(i,j)}$. The* locally testable language in the strict sense *defined by ω (written $LOC_u(\omega)$* [4] *) is the set of pictures $p \in \Sigma^{**}$ such that $B_{i,j}(p) \subseteq \omega$.*

The locally testable language *defined by a finite set of tiles $LOC_{u,eq}(\{\omega_1, \omega_2, \ldots, \omega_n\})$* [5] *is the set of pictures $p \in \Sigma^{**}$ such that for some k, $B_{i,j}(p) = \omega_k$.*

The bordered locally testable language *defined by a finite set of tiles $LOC_{eq}(\{\omega_1, \omega_2, \ldots, \omega_n\})$ is the set of pictures $p \in \Sigma^{**}$ such that for some k, $B_{i,j}(\hat{p}) = \omega_k$.*

---

[3] Maximality as used in [6] is different. It corresponds to the condition $coor(q,p) \not\subseteq coor(q',p)$.

[4] To avoid confusion with LOC defined in [3], we mark these with "u" (stands for *unbordered*, because they do not use boundary symbols).

[5] eq stands for equality test.

Definition 6 Substitution. *If p, q, q' are pictures, $q \trianglelefteq_{(i,j)} p$, and q, q' have the same size, then $p[q'/q]_{(i,j)}$ denotes the picture obtained by replacing the occurrence of q at position (i, j) in p with q'.*

Definition 7 *The (vertical)* mirror image *and the (clockwise)* rotation *of a picture p (with $|p| = (h, k)$), respectively, are defined as follows:*

$$Mirror(p) = \begin{matrix} p(h,1) & \dots & p(h,k) \\ \vdots & \ddots & \vdots \\ p(1,1) & \dots & p(1,k) \end{matrix} \qquad p^R = \begin{matrix} p(h,1) & \dots & p(1,1) \\ \vdots & \ddots & \vdots \\ p(h,k) & \dots & p(1,k) \end{matrix}$$

Notice that the sizes of $Mirror(p)$ and p^R are respectively (h, k) and (k, h).

3 Tile Rewriting Grammars

The main definition follows.

Definition 8 *A* Tile Rewriting Grammar (in short Grammar) *is a tuple (Σ, N, S, R), where Σ is the* terminal *alphabet, N is a set of* nonterminal *symbols, $S \in N$ is the* starting symbol*, R is a set of* rules.

R may contain two kinds of rules:

Fixed size: *$A \to t$, where $A \in N$, $t \in (\Sigma \cup N)^{(h,k)}$, with $h, k > 0$;*
Variable size: *$A \to \omega$, where $A \in N$, $\omega \subseteq (\Sigma \cup N)^{(h,k)}$, with $1 \leq h, k \leq 2$.*

Intuitively a fixed size rule is intended to match a subpicture of (small) bounded size, identical to the right part t. A variable size rule matches any subpicture of any size which can be tiled using *all* the elements t of the tile set ω. However, fixed size rules are not a special case of variable size rules.

Definition 9 *Consider a grammar $G = (\Sigma, N, S, R)$, let $p, p' \in (\Sigma \cup N)^{(h,k)}$ be pictures of identical size, and let γ, γ' be equivalence relations over $coor(p)$.*

We say that (p', γ') derives in one step *from* (p, γ)*, written*

$$(p, \gamma) \Rightarrow_G (p', \gamma')$$

iff for some $A \in N$ and for some rule $\rho : A \to \ldots \in R$ there exists in p a A-subpicture $r \trianglelefteq_{(m,n)} p$, maximal with respect to γ, such that:

- *p' is obtained substituting r with a picture s, that is*

$$p' = p[s/r]_{(m,n)}$$

where s is defined as follows:
Fixed size: *if $\rho = A \to t$, then $s = t$;*
Variable size: *if $\rho = A \to \omega$, then $s \in LOC_{u,eq}(\omega)$.*
- *Let z be $coor_{(m,n)}(r, p)$. Let Γ be the γ-equivalence class containing z. Then, γ' is equal to γ, for all the equivalence classes $\neq \Gamma$; Γ in γ' is divided in two equivalence classes, z and its complement with respect to Γ ($= \emptyset$ if $z = \Gamma$).*
  More formally:

$$\gamma' = \gamma \setminus \{((x_1, y_1), (x_2, y_2)) \mid (x_1, y_1) \in z \text{ xor } (x_2, y_2) \in z\}$$

The subpicture r is named the application area *of rule ρ in the derivation step.*

We say that (q, γ') is derivable from *(p, γ) in n steps, written $(p, \gamma) \stackrel{n}{\Rightarrow}_G (q, \gamma')$, iff $p = q$ and $\gamma = \gamma'$, when $n = 0$, or there are a picture r and an equivalence relation γ'' such that $(p, \gamma) \stackrel{n-1}{\Longrightarrow}_G (r, \gamma'')$ and $(r, \gamma'') \Rightarrow_G (q, \gamma')$. We use the abbreviation $(p, \gamma) \stackrel{*}{\Rightarrow}_G (q, \gamma')$ for a derivation with $n \geq 0$ steps.*

Definition 10 *The* picture language *defined by a grammar G (written $L(G)$) is the set of $p \in \Sigma^{**}$ such that, if $|p| = (h, k)$, then*

$$\left(S^{(h,k)}, coor(p) \times coor(p)\right) \stackrel{*}{\Rightarrow}_G (p, \gamma) \tag{1}$$

where the relation γ is arbitrary. For short we write $S \stackrel{*}{\Rightarrow}_G p$.

Notice that the derivation starts with a S-picture isometric with the terminal picture to be generated, and with the universal equivalence relation over the coordinates. The equivalence relations computed by each step of (1) are called *geminal* relations. When writing examples by hand, it is convenient to visualize the equivalence classes of a geminal relation, by appending the same numerical subscript to the pixels of the application area rewritten by a derivation step. The final classes of equivalence represent in some sense a two dimensional generalization of the parenthesis structure that parenthesized context-free string grammars assign to a sentence.

Example 11 Chinese boxes. *$G = (\Sigma, N, S, R)$, where $\Sigma = \{\ulcorner, \urcorner, \llcorner, \lrcorner, \circ\}$, $N = \{S\}$, and R consists of one fixed size, one variable size rule:*

$$S \to \begin{array}{cc} \ulcorner & \urcorner \\ \llcorner & \lrcorner \end{array}; \quad S \to \left\{ \begin{array}{ccccccccc} \ulcorner & \circ & \circ & S & S & \circ & S & \circ & S & \circ \\ \circ & S, & \llcorner & \circ, & \circ & S, & \circ & \circ, & S & S, & S & S, & S & \circ, & S & \circ, & \circ & \lrcorner \end{array} \right\}$$

For brevity and readability, we will often specify a set of tiles by a sample picture exhibiting the tiles as its subpictures. We write | to separate alternative right parts of rules with the same left part (analogously to string grammars). The previous

grammar becomes:

$$S \to \begin{matrix} \ulcorner & \urcorner \\ \llcorner & \lrcorner \end{matrix} \mid \left[\!\!\left[\begin{matrix} \ulcorner & \circ & \circ & \urcorner \\ \circ & S & S & \circ \\ \circ & S & S & \circ \\ \llcorner & \circ & \circ & \lrcorner \end{matrix} \right]\!\!\right]$$

A picture in $L(G)$ is: $\begin{matrix} \ulcorner & \circ & \circ & \circ & \urcorner \\ \circ & \ulcorner & \circ & \circ & \urcorner & \circ \\ \circ & \circ & \ulcorner & \urcorner & \circ & \circ \\ \circ & \circ & \llcorner & \lrcorner & \circ & \circ \\ \circ & \llcorner & \circ & \circ & \lrcorner & \circ \\ \llcorner & \circ & \circ & \circ & \lrcorner \end{matrix}$ *and is obtained applying the variable size rule twice and then the fixed size rule. We show a complete derivation for a more general version of this language in the following example.*

Example 12 2D Dyck analogue. *The next language L_{box}, a superset of Chinese boxes, can be defined by a sort of blanking rule. But since terminals cannot be deleted without shearing the picture, we replace them with a character b (blank or background).*

Empty frame: *Let $k \geq 0$. An* empty frame *is a picture defined by the regular expression: $(\ulcorner \oplus (\circ)^{k\oplus} \oplus \urcorner) \ominus (\circ \oplus b^{k\oplus} \oplus \circ)^{k\ominus} \ominus (\llcorner \oplus (\circ)^{k\oplus} \oplus \lrcorner)$, i.e. a box bordered by $\circ$, containing just b's.*

Blanking: *The* blanking *of an empty frame p is the picture $del(p)$ obtained by applying the projection $del(x) = b, x \in \Sigma \cup \{b\}$.*

A picture p is in L_{box} iff by repeatedly applying del to subpictures which are empty frames, an empty frame is obtained.

To obtain the grammar, we add the following rules to the Chinese boxes grammar:

$$S \to \left[\!\!\left[\begin{matrix} S & S & X & X \\ S & S & X & X \end{matrix} \right]\!\!\right] \mid \left[\!\!\left[\begin{matrix} S & S \\ S & S \\ X & X \\ X & X \end{matrix} \right]\!\!\right]; \quad X \to \left[\!\!\left[\begin{matrix} S & S \\ S & S \end{matrix} \right]\!\!\right]$$

To illustrate, in Figure 1 we list the derivation steps of a picture. Nonterminals in the same equivalence class are marked with the same subscript.

Although this language can be viewed as a 2D analogue of a Dyck's string language, variations are possible and we do not claim the same algebraic properties as in 1D.

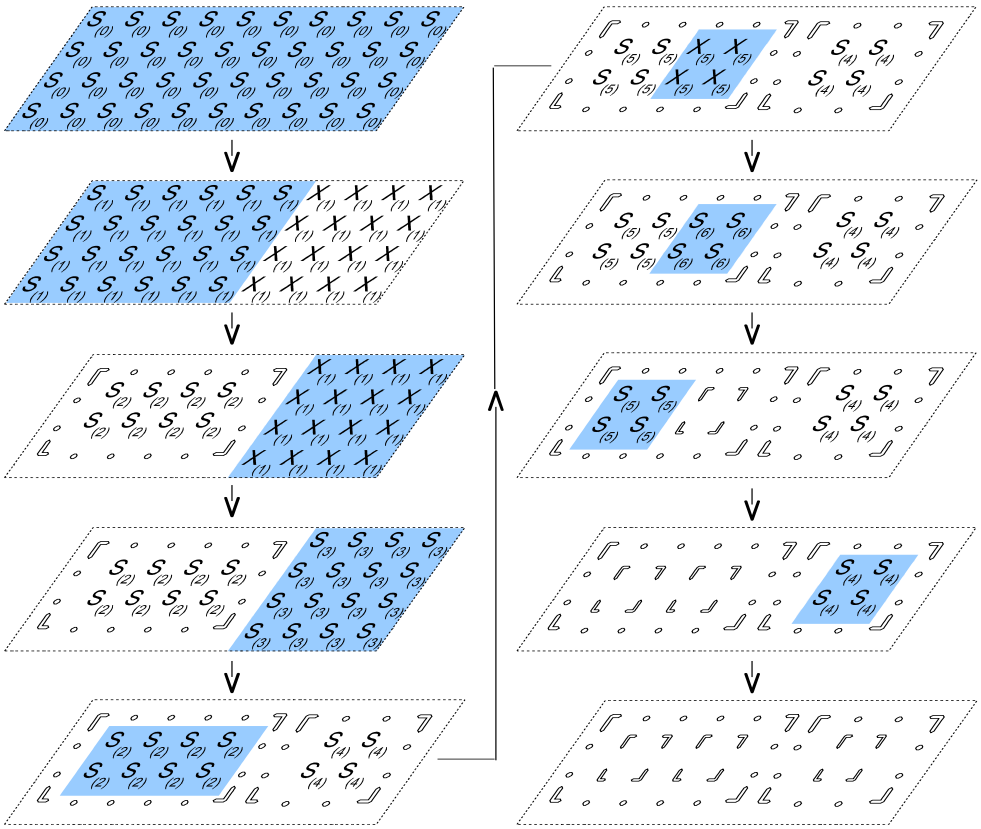


Fig. 1. Example derivation with marked application areas.

3.1 Basic properties

The next two statements, which follow immediately from Definitions 3 and 9, may be viewed as a 2D formulation of well known properties of 1D CF derivations.

Let $p_1 \Rightarrow \ldots \Rightarrow p_{n+1}$ be a derivation, and $r_1 \trianglelefteq_{(i_1,j_1)} p_1, \ldots, r_n \trianglelefteq_{(i_n,j_n)} p_n$ the corresponding application areas.

Disjointness of application areas: For any $p_f, p_g, f < g$, one of the following holds:

(1) $coor_{(i_g,j_g)}(r_g, p_g) \subseteq coor_{(i_f,j_f)}(r_f, p_f)$;
(2) $coor_{(i_f,j_f)}(r_f, p_f) \cap coor_{(i_g,j_g)}(r_g, p_g) = \emptyset$.

That is, the application area of a later step is either totally placed within the application area of a previous step, or it does not overlap. As a consequence, a derivation can be represented in 3D as a well-nested forest of rectangular prisms, the analogous of derivation trees of string languages.

Canonical derivation: The previous derivation is *lexicographic* iff $f < g$ implies $(i_f, j_f) \leq_{lex} (i_g, j_g)$ (where $\leq_{lex}$ is the usual lexicographic order). Then, the following result holds:

$$L(G) \equiv \{p \mid S \stackrel{*}{\Rightarrow}_G p \text{ and } \stackrel{*}{\Rightarrow}_G \text{ is a lexicographic derivation}\}$$

Definition 13 *A rule ρ of a grammar G is* useful *if there exists a derivation $S \stackrel{*}{\Rightarrow}_G p \in \Sigma^{**}$ which makes use of ρ at some step; otherwise ρ is called* useless.

Definition 14 *Consider a grammar $G = (\Sigma, N, S, R)$. A variable size rule $A \to \omega$ is called* concave *iff ω contains an element of the following set:*

$$\left\{ \begin{array}{cccc} A & A & x & A \\ x & A, & A & A, \end{array} \quad \begin{array}{cccc} A & x & A & A \\ A & A, & A & x \end{array} \right\}$$

where $A \in N, x \in N \cup \Sigma, x \neq A$.

Theorem 15 *A concave rule is useless.*

PROOF. By contradiction, if $A \to \omega$, a concave rule, is used in a derivation, then $LOC_{u,eq}$ in Definition 9 compels the use of every tile in ω. But concave tiles generate pictures having a concave area filled with the same nonterminal, say A, and the geminal relation updated by the derivation step is such that this whole area is in the same equivalence class. But Definition 3 makes it impossible to find at following steps, a A-subpicture which is maximal with respect to the geminal relation; hence the derivation fails to produce a terminal picture. □

A useful grammar transformation consists of moving terminal symbols to fixed size rules.

Definition 16 *A grammar G is in* terminal normal form *iff the only rules with terminals have the form $A \to x, x \in \Sigma$, i.e. they are unitary rules.*

Theorem 17 *Every grammar $G = (\Sigma, N, S, R)$ has an equivalent grammar $G' = (\Sigma, N', S, R')$ in terminal normal form.*

PROOF. To construct G', we eliminate terminals from variable size rules and non-unitary fixed size rules. N' contains N, and for every terminal a, we have in N' two nonterminals $\langle a, 0\rangle$ and $\langle a, 1\rangle$. The idea is to replace every homogeneous a-subpicture with a chequered area of $\langle a, 0\rangle$ and $\langle a, 1\rangle$, in which every application area has size $(1, 1)$.

Let $Ch_0^{(m,n)}$ (and $Ch_1^{(m,n)}$) be a chequerboard made of 0 and 1 symbols, starting with a 0 (1, resp.) at the top-leftmost position. Let $\pi : N' \cup (N \times \{0,1\}) \to N'$ be the projection defined as $\pi(\langle a,k\rangle) = \langle a,k\rangle$, if $a \in \Sigma$; $\pi(\langle A,k\rangle) = A$, if $A \in N$.

The mapping $Chequer : \mathscr{P}\left((\Sigma \cup N)^{(m,n)}\right) \to \mathscr{P}\left((N')^{(m,n)}\right)$ is defined as:

$$Chequer(\omega) = \left\{\pi(t \otimes t') \mid t \in \omega \wedge t' \in \{Ch_0^{|t|}, Ch_1^{|t|}\}\right\}$$

Then, for every variable size rule $X \to \omega$ in G, the following rules are in G':

$$\left\{X \to \omega' \mid \omega' \subseteq Chequer(\omega) \wedge Chequer^{-1}(\omega') = \omega\right\}$$

For every non-unitary fixed size rule $X \to t$, the rule $X \to \pi\left(t \otimes Ch_0^{|t|}\right)$ is in G'. Moreover, the unitary fixed size rules $\langle a,0\rangle \to a, \langle a,1\rangle \to a$ are in G'. G' is by construction in terminal normal form.

By construction, rules in G' maintain the same structure and applicability of rules in G, as far as nonterminals in N are concerned. The only difference resides in derived terminal subpictures, that are replaced in G' by chequered subpictures made of new nonterminals, which maintain information about the terminal symbol originally derivable in G in the same area. The chequered structure of these subpictures contains only unitary application areas. Therefore, starting from these subpictures, and using the unitary terminal rules introduced in R', it is always possible to derive homogeneous terminal subpictures, identical to those derivable from G. □

Example 18 *Terminal normal form of Example 11. It is possible to obtain the equivalent terminal normal form grammar by using the construction presented in Theorem 17. For ease of reading, we write the nonterminals* $\langle a,k\rangle, a \in \Sigma, k \in \{0,1\}$ *as* a_k*. The resulting grammar (without useless rules) is the following:*

$$S \to \begin{array}{cc} \ulcorner_0 & \urcorner_1 \\ \llcorner_1 & \lrcorner_0 \end{array} \mid \left[\!\!\left[\begin{array}{cccc} \ulcorner_0 & \circ_1 & \circ_0 & \urcorner_1 \\ \circ_1 & S & S & \circ_0 \\ \circ_0 & S & S & \circ_1 \\ \llcorner_1 & \circ_0 & \circ_1 & \lrcorner_0 \end{array}\right]\!\!\right] \mid \left[\!\!\left[\begin{array}{cccccc} \ulcorner_0 & \circ_1 & \circ_0 & \circ_1 & \circ_0 & \urcorner_1 \\ \circ_1 & S & S & S & S & \circ_0 \\ \circ_0 & S & S & S & S & \circ_1 \\ \circ_1 & S & S & S & S & \circ_0 \\ \circ_0 & S & S & S & S & \circ_1 \\ \llcorner_1 & \circ_0 & \circ_1 & \circ_0 & \circ_1 & \lrcorner_0 \end{array}\right]\!\!\right]$$

$$\ulcorner_0 \to \ulcorner;\ \urcorner_1 \to \urcorner;\ \circ_0 \to \circ;\ \circ_1 \to \circ$$

3.2 Closure Properties

For simplicity, in the following theorem we suppose that $L(G_1), L(G_2)$ contain pictures of size at least (2,2).

Theorem 19 *The family $\mathcal{L}(TRG)$ is closed under union, column/row concatenation, column/row closure operations, rotation, and alphabetical mapping (or projection).*

PROOF. Consider two grammars $G_1 = (\Sigma, N_1, A, R_1)$ and $G_2 = (\Sigma, N_2, B, R_2)$. Suppose for simplicity that $N_1 \cap N_2 = \emptyset$, $S \notin N_1 \cup N_2$, and that G_1, G_2 generate pictures having size at least $(2, 2)$. Then it is easy to show that the grammar $G = (\Sigma, N_1 \cup N_2 \cup \{S\}, S, R_1 \cup R_2 \cup R)$, where

Union $\cup$:

$$R = \left\{ S \to \left[\!\!\left[\begin{matrix} A & A \\ A & A \end{matrix} \right]\!\!\right], \; S \to \left[\!\!\left[\begin{matrix} B & B \\ B & B \end{matrix} \right]\!\!\right] \right\}$$

is such that $L(G) = L(G_1) \cup L(G_2)$.

Concatenation $\oplus / \ominus$:

$$R = \left\{ S \to \left[\!\!\left[\begin{matrix} A & A & B & B \\ A & A & B & B \end{matrix} \right]\!\!\right] \right\}$$

is such that $L(G) = L(G_1) \oplus L(G_2)$. The row concatenation case is analogous.

Closures ${}^{*\oplus}/{}^{*\ominus}$:

$$G = (\Sigma, N_1 \cup \{S\}, S, R_1 \cup R)$$

where

$$R = \left\{ S \to \left[\!\!\left[\begin{matrix} A & A & S & S \\ A & A & S & S \end{matrix} \right]\!\!\right] \mid \left[\!\!\left[\begin{matrix} A & A \\ A & A \end{matrix} \right]\!\!\right] \right\}$$

is such that $L(G) = L(G_1)^{*\oplus}$. The row closure case is analogous.

Rotation R : Construct the grammar $G = (\Sigma, N, A, R')$, where R' is such that, if $B \to t \in R_1$ is a fixed size rule, then $B \to t^R$ is in R'; if $B \to \omega \in R_1$ is a variable size rule, then $B \to \omega'$ is in R', with $t \in \omega$ imply $t^R \in \omega'$. It is easy to verify that $L(G) = L(G_1)^R$.

Projection π : Without loss of generality, we suppose G_1 in terminal normal form (Theorem 17). Consider a projection $\pi : \Sigma_1 \to \Sigma_2$. It is immediate to build a grammar $G' = (\Sigma_2, N_1, A, R_2)$, such that $L(G') = \pi(L(G_1))$: simply apply π to unitary rules. That is, if $X \to x \in R_1$, then $X \to \pi(x) \in R_2$, while the other rules of G_1 remain in R_2 unchanged. □

4 Comparison with other models

We first compare with CF string grammars, then Tiling Systems, and finally with Matz's 2D CF grammars.

4.1 String grammars

If in Definition 8 we choose $h = 1$, then a TRG defines a string language. Such 1D TRG's are easily proved to be equivalent to CF string grammars [6] . In fact, the TRG model for string languages is tantamount to a notational variant [6] of classical CF grammars, where the right parts of rules are local languages.

4.2 Tiling Systems and 2D CF Grammars

The next comparison has to face two technical difficulties: TS are defined by local languages with boundary symbols, which are not present in TRG; and the test of which tiles are present uses inclusion in TS, equality in TRG. First we prove that a class of local languages is strictly included in $\mathcal{L}(TRG)$.

Lemma 20

$$\mathcal{L}(LOC_{u,eq}) \subseteq \mathcal{L}(TRG)$$

PROOF. Consider a local two-dimensional language over Σ defined (without boundaries) by the set of sets of allowed tiles $\{\vartheta_1, \vartheta_2, \ldots, \vartheta_n\}, \vartheta_i \subseteq \Sigma^{(2,2)}$. An equivalent grammar is $S \to \vartheta_1 \mid \vartheta_2 \mid \ldots \mid \vartheta_n$. □

To simplify the comparison with TS, we reformulate them using the terms of Definition 5, showing their equivalence, then we prove strict inclusion with respect to TRG. First we recall the original definition.

Definition 21 *(Definition 7.2 of [3]) A* tiling system *(TS) is a 4-ple $\mathcal{T} = (\Sigma, \Gamma, \vartheta, \pi)$, where Σ and Γ are two finite alphabets,*

(1) ϑ is a finite set of tiles over the alphabet $\Gamma \cup \{\#\}$,

and $\pi : \Gamma \to \Sigma$ is a projection.

Definition 22 *The tiling systems TS_{eq} and $TS_{u,eq}$ are the same as a TS, with the following respective changes:*

- *Replace the local language defined by (1) with $LOC_{eq}(\{\vartheta_1, \vartheta_2, \ldots, \vartheta_n\})$, where ϑ_i is a finite set of tiles over Γ.*
- *Replace the local language defined by (1) with $LOC_{u,eq}(\{\vartheta_1, \vartheta_2, \ldots, \vartheta_n\})$, where ϑ_i is a finite set of tiles over Γ. In $TS_{u,eq}$ there is no boundary symbol #.*

[6] However the empty string cannot be generated by a 1D TRG.

Lemma 23 $\mathcal{L}(TS_{eq}) \equiv \mathcal{L}(TS)$.

PROOF. First, $\mathcal{L}(TS) \subseteq \mathcal{L}(TS_{eq})$. This is easy, because if we consider the tile set ϑ of a TS, by taking $\{\vartheta_1, \vartheta_2, \ldots, \vartheta_n\} = \mathscr{P}(\vartheta)$ (the powerset) we obtain an equivalent TS_{eq}. Second, we have to prove that $\mathcal{L}(TS_{eq}) \subseteq \mathcal{L}(TS)$. In [3], the family of languages $\mathcal{L}(LOC_{eq}(\Omega))$, where Ω is a set of sets of tiles, is proved to be a proper subset of $\mathcal{L}(TS)$ (Theorem 7.8). But $\mathcal{L}(TS)$ is closed with respect to projection, and $\mathcal{L}(TS_{eq})$ is the closure with respect to projection of $\mathcal{L}(LOC_{eq}(\Omega))$. Therefore, $\mathcal{L}(TS_{eq}) \subseteq \mathcal{L}(TS)$. □

Next we prove that boundary symbols can be removed.

Lemma 24 $\mathcal{L}(TS_{u,eq}) \equiv \mathcal{L}(TS_{eq})$.

HINT OF THE PROOF. Part $\mathcal{L}(TS_{eq}) \subseteq \mathcal{L}(TS_{u,eq})$. Let $T = (\Sigma, \Gamma, \{\vartheta_1, \vartheta_2, \ldots, \vartheta_n\}, \pi)$ be a TS_{eq}. For every tile set ϑ_i, separate its tiles containing the boundary symbol # (call this subset ϑ'_i) from the other tiles (ϑ''_i). That is, $\vartheta_i = \vartheta'_i \cup \vartheta''_i$. Introduce a new alphabet Γ' and a bijective mapping $br : \Gamma \to \Gamma'$. We use symbols in Γ' to encode boundary, and new tile sets δ_i to contain them: for every tile t in ϑ''_i, if there is a tile in ϑ'_i which overlaps with t, then encode this boundary in a new tile t' and put it in the set δ_i. For example, suppose $\begin{smallmatrix} a & b \\ c & d \end{smallmatrix} \in \vartheta''_1$ overlaps with $\begin{smallmatrix} \# & \# \\ a & b \end{smallmatrix} \in \vartheta'_1$ and with $\begin{smallmatrix} d & \# \\ \# & \# \end{smallmatrix} \in \vartheta'_1$, then both $\begin{smallmatrix} br(a) & br(b) \\ c & d \end{smallmatrix}$, and $\begin{smallmatrix} a & br(b) \\ br(c) & br(d) \end{smallmatrix}$ are in δ_i.

Consider a $TS_{u,eq}$ $T' = (\Sigma, \Gamma \cup \Gamma', \Omega, \pi')$, where π' extends π to Γ' as follows: $\pi'(br(a)) = \pi'(a) = \pi(a), a \in \Gamma$, and $ubr : \Gamma \cup \Gamma' \to \Gamma$ is defined as $ubr(a) = br^{-1}(a)$, if $a \in \Gamma'$, otherwise $= a$, and it is naturally extended to tiles and tile sets. Ω is the set:

$$\{\vartheta \mid \vartheta \subseteq \vartheta''_i \cup \delta_i \wedge ubr(\vartheta) = \vartheta''_i \wedge \vartheta \cap \delta_i \neq \emptyset \wedge 1 \leq i \leq n\}.$$

The proof that $L(T) = L(T')$ is straightforward and is omitted.

Part $\mathcal{L}(TS_{u,eq}) \subseteq \mathcal{L}(TS_{eq})$. Let $T = (\Sigma, \Gamma, \{\vartheta_1, \vartheta_2, \ldots, \vartheta_n\}, \pi)$ be a $TS_{u,eq}$. To construct an equivalent TS_{eq}, we introduce the boundary tile sets δ_i, defined as follows. For every tile $\begin{smallmatrix} a & b \\ c & d \end{smallmatrix} \in \vartheta_i$, the following tiles are in δ_i:

$$\left\{ \begin{smallmatrix} \# & \# \\ \# & a \end{smallmatrix}, \begin{smallmatrix} \# & \# \\ a & b \end{smallmatrix}, \begin{smallmatrix} \# & \# \\ b & \# \end{smallmatrix}, \begin{smallmatrix} \# & a \\ \# & c \end{smallmatrix}, \begin{smallmatrix} b & \# \\ d & \# \end{smallmatrix}, \begin{smallmatrix} \# & c \\ \# & \# \end{smallmatrix}, \begin{smallmatrix} c & d \\ \# & \# \end{smallmatrix}, \begin{smallmatrix} d & \# \\ \# & \# \end{smallmatrix} \right\}$$

Consider a TS_{eq} $T' = (\Sigma, \Gamma, \Omega, \pi)$, where Ω is the set:

$$\{\vartheta \cup \vartheta_i \mid \vartheta \subseteq \delta_i \wedge \vartheta \neq \emptyset \wedge 1 \leq i \leq n\}\,.$$

It is easy to show that $L(T) = L(T')$. □

Example 7.2 of [3], the language of squares over the alphabet $\{a\}$, is defined by the following $TS_{u,eq}$:

$$\vartheta_1 = \left[\!\!\left[\begin{matrix} 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 3 \end{matrix}\right]\!\!\right] ;\ \vartheta_2 = \left[\!\!\left[\begin{matrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{matrix}\right]\!\!\right] ;\ \vartheta_3 = \left[\!\!\left[\begin{matrix} 1 & 0 \\ 0 & 3 \end{matrix}\right]\!\!\right]$$

$$\pi(0) = \pi(1) = \pi(2) = \pi(3) = a$$

Theorem 25

$$\mathcal{L}(TS) \subseteq \mathcal{L}(TRG)$$

PROOF. It follows from Theorems 19, 20, 23, 24, and the fact that $\mathcal{L}(TS_{u,eq})$ is the closure of $\mathcal{L}(LOC_{u,eq})$ with respect to projection. □

The following strict inclusion is an immediate consequence of the fact that, for 1D languages, $\mathcal{L}(TS) \subset \mathcal{L}(CF)$, and $\mathcal{L}(TRG) = \mathcal{L}(CF) \setminus \{\epsilon\}$. But we prefer to prove it by exhibiting an interesting picture language, made by the vertical concatenation of two specularly symmetrical rectangles.

Theorem 26

$$\mathcal{L}(TS) \neq \mathcal{L}(TRG)$$

PROOF. Let $\Sigma = \{a, b\}$. Consider the 2D language of palindromic columns, such as $\begin{matrix} a & b & b \\ b & a & b \\ b & a & b \\ a & b & b \end{matrix}$: $L = \{p \mid p = s \ominus Mirror(s) \wedge s \in \Sigma^{(h,k)}, h > 1, k \geq 1\}$.

Consider the grammar G:

$$S \rightarrow \left[\!\!\left[\begin{matrix} X & S & S \\ X & S & S \end{matrix}\right]\!\!\right] \mid \left[\!\!\left[\begin{matrix} X & S \\ X & S \end{matrix}\right]\!\!\right] \mid \left[\!\!\left[\begin{matrix} X \\ X \end{matrix}\right]\!\!\right]$$

$$X \rightarrow \begin{matrix} a \\ a \end{matrix} \mid \begin{matrix} b \\ b \end{matrix} \mid \left[\!\!\left[\begin{matrix} a \\ X \\ X \\ a \end{matrix} \right]\!\!\right] \mid \left[\!\!\left[\begin{matrix} b \\ X \\ X \\ b \end{matrix} \right]\!\!\right]$$

It is easy to see that $L(G) = L$.
We prove by contradiction that $L \notin \mathcal{L}(TS)$. Suppose that $L \in \mathcal{L}(TS)$, therefore L is a projection of a local language L' defined over some alphabet Γ. Let $a = |\Sigma|$ and $b = |\Gamma|$, with $a \leq b$. For an integer n, let:

$$L_n = \{p \mid p = s \ominus Mirror(s) \wedge |s| = (n, n)\}.$$

Clearly, $|L_n| = a^{n^2}$. Let L'_n be the set of pictures in L' over Γ whose projections are in L_n. By choice of b and by construction of L_n there are at most b^n possibilities for the n-th and $(n+1)$-th rows in the pictures of L'_n, because this is the number of mirrored stripe pictures of size $(2, n)$ over Γ.

For n sufficiently large $a^{n^2} \geq b^n$. Therefore, for such n, there will be two different pictures $p = s_p \ominus Mirror(s_p), q = s_q \ominus Mirror(s_q)$ such that the corresponding $p' = s'_p \ominus s''_p, q' = s'_q \ominus s''_q$ have the same n-th and $(n+1)$-th rows. This implies that, by definition of local language, pictures $v' = s'_p \ominus s''_q, w' = s'_q \ominus s''_p$ belong to L'_n, too. Therefore, pictures $\pi(v') = s_p \ominus Mirror(s_q)$, and $\pi(w') = s_q \ominus Mirror(s_p)$ belong to L_n. But this is a contradiction. □

We terminate by comparing with a different generalization of CF grammars in two dimensions, Matz's CF Picture Grammars ($CFPG$)[5], a model syntactically very similar to string CF grammars. The main difference is that the right parts of their rules use $\oplus, \ominus$ operators. Nonterminals denote unbound rectangular pictures. Derivation is analogous to string grammars, but the resulting regular expression may or may not define a picture (e.g. $a \oplus (b \ominus b)$ does not generate any picture).

Theorem 27

$$\mathcal{L}(CFPG) \subseteq \mathcal{L}(TRG)$$

HINT OF THE PROOF. Consider now a Matz's CFPG grammar in Chomsky Normal Form. It may contain three types of rules: $A \rightarrow B \oplus C;\ A \rightarrow B \ominus C;\ A \rightarrow a$. Moreover, suppose that $B \neq C$ (this is always possible, if we permit copy rules like $A \rightarrow B$). Then, $A \rightarrow B \oplus C$ corresponds to the following TRG rules:

$$A \rightarrow \left[\!\!\left[\begin{matrix} B & B & C & C \\ B & B & C & C \end{matrix} \right]\!\!\right] \mid \left[\!\!\left[\begin{matrix} B & C & C \\ B & C & C \end{matrix} \right]\!\!\right] \mid \left[\!\!\left[\begin{matrix} B & B & C \\ B & B & C \end{matrix} \right]\!\!\right] \mid$$

$$\left[\!\!\left[B\ B\ C\ C \right]\!\!\right] \mid \left[\!\!\left[B\ C\ C \right]\!\!\right] \mid \left[\!\!\left[B\ B\ C \right]\!\!\right] \mid B\ C$$

To obtain $A \to B$, just delete C from the previous rules. The $\ominus$ case is analogous to $\oplus$, while $A \to a$ is trivial. □

Theorem 28

$$\mathcal{L}(CFPG) \neq \mathcal{L}(TRG)$$

PROOF. It is a consequence of Theorems 25, 26, and 27, and the fact from [5] that $\mathcal{L}(TS) \not\subseteq \mathcal{L}(CFPG)$. □

An example of a TRG but not CFPG language is the following. We know from [5] that the "cross" language, which consists of two perpendicular b-lines on a background of a, is not in $\mathcal{L}(CFPG)$. It is easy to show that the following grammar defines the language:

$$S \to \left[\!\!\left[\begin{matrix} B & B & A & A \\ B & B & A & A \\ C & C & D & D \\ C & C & D & D \end{matrix} \right]\!\!\right];$$

$$B \to \left[\!\!\left[\begin{matrix} a & a \\ a & a \\ b & b \end{matrix} \right]\!\!\right]; \; A \to \left[\!\!\left[\begin{matrix} b & a & a \\ b & a & a \\ b & b & b \end{matrix} \right]\!\!\right]$$

$$C \to \left[\!\!\left[\begin{matrix} a & a \\ a & a \end{matrix} \right]\!\!\right]; \; D \to \left[\!\!\left[\begin{matrix} b & a & a \\ b & a & a \end{matrix} \right]\!\!\right]$$

The fine control on line connections provided by TRG rules allows the definition of complex recursive patterns, exemplified by the spirals presented in the Appendix.

5 Conclusions

The new TRG model extends the context-free string grammars to two dimensions. Each rule rewrites a homogeneous rectangle as an isometric one, tiled with a specified tile set. In a derivation the rectangles, rewritten at each step, are partially ordered by the subpicture relation, which can be represented in three dimensions by a forest of well nested prisms, the analogue of syntax trees for strings.

Spirals and nested boxes are typical examples handled by TRG.

The generative capacity of TRG is greater than that of two previous models: Tiling Systems and Matz's context free picture grammars.

Practical applicability to picture processing tasks (such as pattern recognition and image compression) remains to be investigated, which will ultimately depend on the expressive power of the new model and on availability of good parsing algorithms.

The analogy with string grammars raises to the educated formal linguist a variety of questions, such as the formulation of a pumping lemma. For comparison with other models, several questions may be considered, e.g whether TRG and TS families coincide on a unary alphabet, or the generative capacity of non-recursive TRG versus TS.

Acknowledgment

Antonio Restivo called our attention to the problem of "2D Dyck languages". We thank Alessandra Cherubini, Pierluigi San Pietro, Alessandra Savelli, and Daniele Scarpazza for their comments.

References

[1] Henning Fernau and Rudolf Freund. Bounded parallelism in array grammars used for character recognition. In Petra Perner, Patrick Wang, and Azriel Rosenfeld, editors, *Advances in Structural and Syntactical Pattern Recognition (Proceedings of the SSPR'96)*, volume 1121, pages 40–49. Springer-Verlag, 1996.

[2] Dora Giammarresi and Antonio Restivo. Recognizable picture languages. *International Journal Pattern Recognition and Artificial Intelligence*, 6(2-3):241–256, 1992. Special Issue on *Parallel Image Processing*.

[3] Dora Giammarresi and Antonio Restivo. Two-dimensional languages. In Arto Salomaa and Grzegorz Rozenberg, editors, *Handbook of Formal Languages*, volume 3, Beyond Words, pages 215–267. Springer-Verlag, Berlin, 1997.

[4] Katsushi Inoue and Akira Nakamura. Some properties of two-dimensional on-line tessellation acceptors. *Information Sciences*, 13:95–121, 1977.

[5] Oliver Matz. Regular expressions and context-free grammars for picture languages. In *14th Annual Symposium on Theoretical Aspects of Computer Science*, volume 1200 of *Lecture Notes in Computer Science*, pages 283–294, Lübeck, Germany, 27 February–March 1 1997. Springer-Verlag.

[6] Stefano Crespi Reghizzi and Matteo Pradella. Tile rewriting grammars. In *7th International Conference on Developments in Language Theory (DLT 2003)*, volume 2710 of *Lecture Notes in Computer Science*, pages 206–217, Szeged, Hungary, July 2003. Springer-Verlag.

[7] Rani Siromoney. Advances in Array Languages. In Hartmut Ehrig, Manfred Nagl, Grzegorz Rozenberg, and Azriel Rosenfeld, editors, *Proc. 3rd Int. Workshop on Graph-Grammars and Their Application to Computer Science*, volume 291 of *Lecture Notes in Computer Science*, pages 549–563. Springer-Verlag, 1987.

6 Appendix

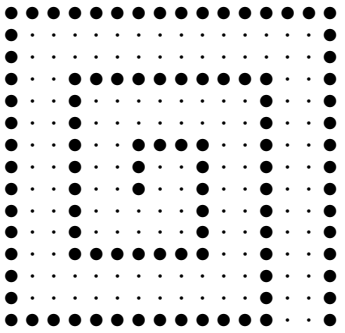
Grammar for defining discrete Archimedean spirals with step 3 [7] .

$$S \to \left[\!\!\left[\begin{array}{ccccccc} A & A & H & H & H & B & B \\ A & A & H & H & H & B & B \\ V & V & Q & Q & Q & W & W \\ V & V & Q & Q & Q & W & W \\ C & C & K & K & \bullet & D & D \\ C & C & K & K & \bullet & D & D \end{array}\right]\!\!\right] ;\ Q \to \left[\!\!\left[\begin{array}{cc} S & S \\ S & S \end{array}\right]\!\!\right] \ | \begin{array}{ccc} \bullet & \bullet & \bullet \\ \bullet & \cdot & \bullet \\ \bullet & \cdot & \bullet \end{array}$$

$$A \to \begin{array}{ccc} \bullet & \bullet & \bullet \\ \bullet & \cdot & \cdot \\ \bullet & \cdot & \cdot \end{array} ;\ B \to \begin{array}{ccc} \bullet & \bullet & \bullet \\ \cdot & \cdot & \bullet \\ \cdot & \cdot & \bullet \end{array} ;\ C \to \begin{array}{ccc} \bullet & \cdot & \cdot \\ \bullet & \cdot & \cdot \\ \bullet & \bullet & \bullet \end{array} ;\ D \to \begin{array}{ccc} \cdot & \cdot & \bullet \\ \cdot & \cdot & \bullet \\ \cdot & \cdot & \bullet \end{array}$$

$$H \to \left[\!\!\left[\begin{array}{cc} \bullet & \bullet \\ \cdot & \cdot \\ \cdot & \cdot \end{array}\right]\!\!\right] ;\ K \to \left[\!\!\left[\begin{array}{cc} \cdot & \cdot \\ \cdot & \cdot \\ \bullet & \bullet \end{array}\right]\!\!\right] ;\ V \to \left[\!\!\left[\begin{array}{cc} \bullet & \cdot \\ \bullet & \cdot \end{array}\right]\!\!\right] ;\ W \to \left[\!\!\left[\begin{array}{cc} \cdot & \bullet \\ \cdot & \bullet \end{array}\right]\!\!\right]$$

An example picture:



[7] By Daniele Paolo Scarpazza.