Politecnico di Milano
Dipartimento di Elettronica e Informazione
Via Ponzio 34 / 5
Milano, 20133, Italy

Daniele Paolo Scarpazza
<scarpazz@elet.polimi.it>

A PARSING TECHNIQUE FOR TILE-REWRITING GRAMMARS

Technical Report 2004.11

A Parsing Technique for Tile Rewriting Picture Grammars

Daniele Paolo Scarpazza

DEI – Politecnico di Milano
Piazza Leonardo da Vinci, 32,
I-20133 Milano, Italy
e-mail: scarpazz@elet.polimi.it

Abstract

Recently, a novel model, called Tile Rewriting Grammar (TRG), has been introduced to apply the generative grammar approach to picture languages, or 2D languages. In many respects, the TRGs can be considered the equivalent of context-free (CF) grammars for 2D languages. However, the possibility to investigate applications was precluded so far by the lack of a good parsing algorithm. We propose a parsing algorithm for TRGs, which can be described as an extension to 2D of Cocke, Kasami and Younger's classical parsing technique for 1D context-free grammars.

1 Introduction

Traditionally, several approaches were conceived to describe picture languages. Authors in [1] relate and compare these approaches, and they identify the REC class as the correspondent of regular languages for 2D languages. For example, they report Giammarresi and Restivo's *Tiling Systems* (TS) [2] and Inoue and Nakamura's *2D Online tessellation Automata* (2OTA) [3] as formalisms which generate languages in REC. More recent attempts to define an analogous of CF languages for pictures were less successful, especially as far as expressibility and theoretical relations with REC are concerned (see for instance [4]).

A new formal model for context-free picture languages, called Tile Rewriting Grammar (TRG) has been recently introduced in [5, 6]. TRGs ideally originate from the results presented in [1], with a twofold aim. First, to define a context-free formalism with theoretical relationships with REC comparable with those between string CF grammars and regular languages. Secondly, to obtain a descriptive device having elegance and descriptive adequacy analogous to CF string grammars.

However, TRGs lacked so far an associated parsing technique, and this made it hard to conceive any applications for them. Moreover, to the best of our knowledge, even for picture grammars (where pictures are defined as 2D arrays of symbols of a finite alphabet) there are no good parsing algorithms in literature. There exist works in the field of visual languages, such as [7], but

they consider a quite different approach to 2D descriptions. An early attempt to parse Rosenfeld's *Array Grammars* [8] is in [9], but it seems to be limited to interactive derivations of pictures.

In this paper we address the problem of parsing TRGs, which is an important step in 2D language research. We believe that the availability of an efficient parsing technique for TRGs will enable a wide class of applications of this descriptive formalism. We propose a parsing approach which follows the classical Cocke, Kasami and Younger's technique [10]: it works bottom-up, and recognizes subpictures as result of application of grammar rules, starting from the simplest subpictures of the input, i.e. its pixels.

This paper is organized as follows. In Section 2 we introduce 2D languages, notations, and Tile Rewriting Grammars. In Section 3 we describe the algorithm. In Section 4 we draw the conclusions. In the appendix we illustrate a complete example of our algorithm recognizing an input picture.

2 Pictures and Grammars

In this section we recall definitions and notations needed for 2D languages, together with the TRG model. Most of them come from [1] and [6].

2.1 Pictures and subpictures

Definition 1. *If a picture p has m rows and n columns, we write that $|p| = (m, n)$, and $|p|_{row} = m, |p|_{col} = n$. Each element $p_{i,j}$ is called a* pixel. *For a finite alphabet Σ, the set of pictures is Σ^{**}. For $m, n \geq 1$, $\Sigma^{(m,n)}$ denotes the set of pictures of size (m, n). If all pixels are identical to $C \in \Sigma$ the picture is said* homogeneous *and denoted as C-picture.*

Definition 2. Subpicture. *Let p be a picture of size (m, n) and q a picture of size (m', n'). We say that q is a* subpicture *of p at position (i, j), and we write:*

$$q \trianglelefteq_{(i,j)} p$$

if $m' \leq m, n' \leq n$ and there exist integers i, j, with $(i \leq m - m' + 1, j \leq n - n' + 1)$, such that $q_{i',j'} = p_{i+i'-1,j+j'-1}$ for all $1 \leq i' \leq m', 1 \leq j' \leq n'$. The shorthand notation $q \trianglelefteq p$ means that $\exists i, j (q \trianglelefteq_{(i,j)} p)$.

Example 1. If $p = \begin{pmatrix} a & d & g & j & m \\ b & e & h & k & n \\ c & f & i & l & o \end{pmatrix}$ and $q = \begin{pmatrix} e & h & k \\ f & i & l \end{pmatrix}$, then: $q \trianglelefteq_{(2,2)} p$.

Definition 3. 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'.*

Example 2. If p and q are as in the previous example, and $q' = \begin{pmatrix} Z & Z & Z \\ Z & Z & Z \end{pmatrix}$, then:

$$p[q'/q]_{(2,2)} = \begin{pmatrix} a & d & g & j & m \\ b & Z & Z & Z & n \\ c & Z & Z & Z & o \end{pmatrix}.$$

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

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

$B_{1,n}$ is defined only on $\Sigma^{(1,*)}$ (horizontal strings), and $B_{m,1}$ only on $\Sigma^{(*,1)}$ (vertical strings). For brevity, for tiles of size $(1,2)$, $(2,1)$, or $(2,2)$, we introduce the following notation:

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

Definition 5. Horizontal Overlapping. *If p and q are pictures of the same size (m,n) and*

$$\forall i,j,\ 1 \le i \le m,\ 2 \le j \le n :\ p_{i,j} = q_{i,j-1},$$

then $p \parallel q$, also called the horizontal overlapping *of p and q, denotes a picture of size $(m, n+1)$ such that:*

$$\forall i,j,\ 1 \le i \le m,\ 1 \le j \le n :\ (p \parallel q)_{i,j} = p_{i,j} \quad \wedge$$

$$\forall i,j,\ 1 \le i \le m,\ 2 \le j \le n+1 :\ (p \parallel q)_{i,j} = q_{i,j-1}.$$

If one of the above conditions is not satisfied, then $p \parallel q$ is not defined.

Definition 6. Vertical Overlapping. *If p and q are pictures of the same size (m,n) and*

$$\forall i,j,\ 2 \le i \le m,\ 1 \le j \le n :\ p_{i,j} = q_{i-1,j},$$

then $\frac{p}{\overline{q}}$, also called the vertical overlapping *of p and q, denotes a picture of size $(m+1, n)$ such that:*

$$\forall i,j,\ 1 \le i \le m,\ 1 \le j \le n :\ \left(\frac{p}{\overline{q}}\right)_{i,j} = p_{i,j} \quad \wedge$$

$$\forall i,j,\ 2 \le i \le m+1,\ 1 \le j \le n :\ \left(\frac{p}{\overline{q}}\right)_{i,j} = q_{i-1,j}.$$

If one of the above conditions is not satisfied, then $\frac{p}{\overline{q}}$ is not defined.

2.2 Rectangles, ceilings and coordinates

Definition 7. Operators $\boxtimes$ and $\boxplus$. *A* coordinate *is a pair of positive integers. Given two positive integers r and c, the notation $r \boxtimes c$ denotes the set of coordinates $\{1, 2, ..., r\} \times \{1, 2, ..., c\}$.*
Given a set of coordinates $C = \{(i_1, j_1), (i_2, j_2), ...\}$ and a coordinate (i, j), the notation $C \boxplus (i, j)$ denotes the following set of coordinates:

$$C \boxplus (i, j) = \{(i_1 + i - 1, j_1 + j - 1), (i_2 + i - 1, j_2 + j - 1), ...\}.$$

Intuitively, operator $\boxplus$ translates coordinates by an offset $(i - 1, j - 1)$.

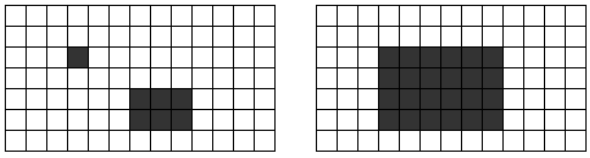
Definition 8. Rectangle. *Any set of coordinates that can be written in the form $r \boxtimes c \boxplus (i, j)$ is said a* rectangle. *The notation $\mathcal{R}(r \boxtimes c)$ indicates the set of all rectangles α such that $\alpha \subseteq r \boxtimes c$.*

Note that $r \boxtimes c \boxplus (i, j)$ contains the coordinates of a subpicture of size (r, c), such that coordinate of top-left pixel is (i, j). For any positive r and c, $r \boxtimes c$ is a rectangle and $r \boxtimes c = r \boxtimes c \boxplus (1, 1)$.

Example 3. $2 \boxtimes 3 = \{(1, 1), (1, 2), (1, 3), (2, 1), (2, 2), (2, 3)\}$ is a rectangle. $2 \boxtimes 3 \boxplus (4, 5) = \{(4, 5), (4, 6), (4, 7), (5, 5), (5, 6), (5, 7)\}$ is a rectangle. An intersection between rectangles can be a rectangle: $2 \boxtimes 3 \boxplus (4, 5) \cap 2 \boxtimes 3 \boxplus (5, 6) = 1 \boxtimes 2 \boxplus (5, 6)$.

Definition 9. Ceiling. *Given a set of coordinates C, the* ceiling *of C, denoted as $\lceil C \rceil$, is the smallest rectangle which is either a superset or equal to C.*

Example 4.



$$\lceil (1 \boxtimes 1 \boxplus (3, 4)) \ \cup \ (2 \boxtimes 3 \boxplus (5, 7)) \rceil =$$
$$= \lceil \{(3, 4), (5, 7), (5, 8), (5, 9), (6, 7), (6, 8), (6, 9)\} \rceil =$$
$$= 4 \boxtimes 6 \boxplus (3, 4)$$

Definition 10. *Given a picture p such that $|p| = (m, n)$ and a rectangle $r \boxtimes c \boxplus (i, j)$, the notation $p[r \boxtimes c \boxplus (i, j)]$ indicates the subpicture q such that $|q| = (r, c)$ and $q \trianglelefteq_{(i,j)} p$.*

Definition 11. *If $q \trianglelefteq_{(i,j)} p$, we define $coor_{(i,j)}(q, p)$ as the set of coordinates of all the pixels of p where q is located: $|q|_{row} \boxtimes |q|_{col} \boxplus (i, j)$. Conventionally, $coor_{(i,j)}(q, p) = \emptyset$ if q is not a subpicture of p. If $q = p$, we write $coor(p)$ instead of $coor_{(1,1)}(p, p)$.*

Example 5. If p and q are the same as in the previous examples,
$coor_{(2,2)}(q, p) = \{(2, 2), (2, 3), (2, 4), (3, 2), (3, 3), (3, 4)\}$.

2.3 Tile Rewriting Grammars

The TRG model combines isometric rewriting rules (like in array grammars [8]) with tiles, defining a locally-testable language. This section recalls the TRG model through its main definitions. For a complete description, see [6].

Definition 12. Locally testable language[1]. *Given a finite set of tiles*

$$\omega = \{t_1, t_2, ...\} \subseteq \Sigma^{(i,j)},$$

the locally testable language *defined over it, indicated as* $LOC(\omega)$*, is the set of pictures* $p \in \Sigma^{**}$ *such that* $B_{i,j}(p) = \omega$*.*

Definition 13. *A* Tile Rewriting Grammar *(TRG, for short) 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* $A \in N$, $t \in (\Sigma \cup N)^{(m,n)}$, *with* $m, n > 0$;
Variable size: $A \to \omega$, *where* $A \in N$, $\omega \subseteq (\Sigma \cup N)^{(m,n)}$, *with* $m, n \in \{1, 2\}$.

The right-hand side of a fixed size rule is a tile; the right-hand side of a variable size rule is a set of tiles. A fixed size rule $A \to t$ rewrites an A-subpicture (isometric with t) as t. A variable size rule $A \to \omega$ rewrites an A-subpicture as one of the pictures which can be tiled using all the tiles in ω.

The next concepts constrain the derivations in order to obtain a 2D analogous of the derivation tree.

Definition 14. Equivalence*. Let* γ *be a given equivalence relation on* $coor(p)$*. We use the notation* $(x, y) \overset{\gamma}{\sim} (x', y')$ *to indicate that two coordinates* (x, y) *and* (x', y') *are equivalent according to* γ*.*
Given two subpictures q, q' *of* p *such that* $q \trianglelefteq_{(i,j)} p, q' \trianglelefteq_{(i',j')} p$*, we say that* q *and* q' *are* γ-equivalent*, and we write* $q \overset{\gamma}{\sim} q'$*, iff for all pairs* $(x, y) \in coor_{(i,j)}(q, p)$ *and* $(x', y') \in coor_{(i',j')}(q', p)$ *it holds* $(x, y) \overset{\gamma}{\sim} (x', y')$*.*

Definition 15. Maximality*. 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*

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

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

Definition 16. Derivation*. Consider a grammar* $G = (\Sigma, N, S, R)$*, let* $p, p' \in (\Sigma \cup N)^{(m,n)}$ *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*

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

[1] Several different notions of local testability appear in literature. See [6] for more on the topic.

iff for some rule $A \to \ldots \in R$ there exists in p a A-subpicture $r \trianglelefteq_{(i,j)} p$, maximal with respect to γ, and p' is obtained substituting r with a picture s, that is

$$p' = p[s/r]_{(i,j)}$$

where s is defined as follows:

Fixed size: *if $A \to t$, then $s = t$;*
Variable size: *if $A \to \omega$, then $s \in LOC(\omega)$.*

Additionally, γ' is obtained from γ as follows. Let z be $coor_{(i,j)}(r, p)$. 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 said the *application area* in the derivation step. We say that (p', γ') is *derivable from* (p, γ) in d steps, written $(p, \gamma) \stackrel{d}{\Rightarrow}_G (p', \gamma')$, iff $p = p'$ and $\gamma = \gamma'$, when $d = 0$, or there are a picture r and an equivalence relation γ'' such that $(p, \gamma) \stackrel{d-1}{\Longrightarrow}_G (r, \gamma'')$ and $(r, \gamma'') \Rightarrow_G (p', \gamma')$. We use the abbreviation $(p, \gamma) \stackrel{*}{\Rightarrow}_G (p', \gamma')$ for a derivation with $d \geq 0$ steps.

Definition 17. *The* picture language *defined by a grammar G (written $\mathcal{L}(G)$) is the set of $p \in \Sigma^{**}$ such that, if $|p| = (m, n)$, then*

$$\left(S^{(m,n)}, coor(p) \times coor(p)\right) \stackrel{*}{\Rightarrow}_G (p, \gamma)$$

for some γ. For short, we write $S \stackrel{*}{\Rightarrow}_G p$.

Note 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.

Example 6. An example TRG grammar is $G = (\Sigma, N, S, R)$, as follows:

$$\Sigma = \{x, o\}$$
$$N = \{S, A, B\}$$
$$R = \left\{ R_1 : S \to \left[\!\!\left[\begin{matrix} A & A & B & B \\ A & A & B & B \end{matrix} \right]\!\!\right], R_2 : A \to \left[\!\!\left[\begin{matrix} x & x & x \\ x & o & o \\ x & o & o \\ x & x & x \end{matrix} \right]\!\!\right], R_3 : B \to \left[\!\!\left[\begin{matrix} x & x & x \\ o & o & x \\ o & o & x \\ x & x & x \end{matrix} \right]\!\!\right] \right\}$$

This grammar generates the language of rectangles of o symbols framed by x symbols (like the picture portrayed below). This language is local, nevertheless we introduce it because it is convenient to illustrate the algorithm, as reported in the appendix.

$$p = \begin{pmatrix} x & x & x & x & x & x \\ x & o & o & o & o & x \\ x & o & o & o & o & x \\ x & o & o & o & o & x \\ x & x & x & x & x & x \end{pmatrix}$$

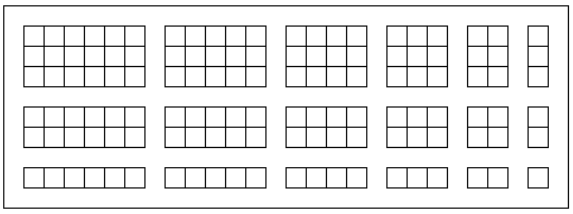
3 Parsing TRGs

3.1 Definitions

In this section we introduce the notions of tableau, multipicture, monopicture and compatibility, which play an important role as data structures used by our parsing algorithm. In particular, the tableau plays the same role as the recognition table in CKY.

Definition 18. Tableau. *A* tableau T *of size* (m, n) *is a matrix containing* $m \cdot n$ *matrices, each denoted as* $T_{r,c}, 1 \leq r \leq m, 1 \leq c \leq n$. *Matrix* $T_{r,c}$ *has size* $(m - r + 1, n - c + 1)$. *The notation* $T[r \boxtimes c \boxplus (i, j)]$ *indicates the* (i, j) *element of matrix* $T_{r,c}$, *or* $(T_{r,c})_{i,j}$. *Matrices inside* T *are called* cells. *Cells inside those matrices are called* elementary cells.

Example 7. The first cell $T_{1,1}$ in a tableau T of size (m, n) is a matrix which also has size (m, n). Cell $T_{1,2}$ has one column less, and so on. Cell $T_{1,n}$ has one column. Cell $T_{2,1}$ has one row less. Cell $T_{n,1}$ has one row. The last cell $T_{m,n}$ has size (1,1). The following figure shows an empty $(3, 6)$–tableau.



A tableau has exactly one *cell* for each of the possible sizes of a rectangle of size smaller or equal than (m, n). More precisely, a tableau has exactly one *elementary cell* for each such rectangle included in rectangle (m, n). Therefore the number of elementary cells in a tableau is equal to:

$$|\mathcal{R}(m \boxtimes n)| = \frac{m(m-1)}{2} \frac{n(n-1)}{2} = \frac{m(m-1)n(n-1)}{4}.$$

Our algorithm will use one tableau to store candidates, devices used to mark partially-recognized rules. Their definition follows.

Definition 19. Candidate. *Given a TRG grammar $G = (\Sigma, N, S, R)$, and a picture $p \in (\Sigma \cup N)^{(m,n)}$, a* candidate *is a quadruple $(R_e, \omega_x, \alpha, s)$ such that $R_e \in R$, $R_e = A \to \omega$, $\omega_x \subset \omega$, $\alpha \subseteq \mathcal{R}(m \boxtimes n)$ and $s \trianglelefteq p$. A candidate $(R_e, \emptyset, \alpha, s)$ is said* complete.

At a given time, in a tableau elementary cell there are zero or more candidates. When an elementary cell $T[r \boxtimes c \boxplus (i,j)]$ contains a candidate (R_e, ω_x, α), it means that rectangle $r \boxtimes c \boxplus (i,j)$ contains only tiles which appear in the right-hand side of rule R_e. Possibly, not all the tiles in the right-hand side of R_e appear in the rectangle; the missing ones are indicated in ω_x. α is called the *scope* of symbols inside the rectangle, and it encodes application areas.

Definition 20. Multipicture, monopicture. *Given an alphabet Σ, consider the set $\Psi = \Sigma \times \mathcal{R}(m \boxtimes n)$. A* monopicture *over Σ is a matrix M of $m \times n$ cells, where each cell contains an element from Ψ. A* multipicture *over Σ is a matrix $\mathcal{M}$ of $m \times n$ cells, where each cell contains a set of elements from Ψ.*

In simple words, each cell of a monopicture M of size (m,n) over alphabet Σ contains exactly one pair (symbol, rectangle) such as (A, α), where $A \in \Sigma$, and α is a rectangle contained in $m \boxtimes n$. We call α the *scope* of symbol A. In multipictures, each cell contains zero or more such pairs.

Example 8. p and p' are pictures, and $\mathcal{M}$ is a multipicture.

$$p = \begin{pmatrix} x & x & x & x & x & x \\ x & o & o & o & o & x \\ x & o & o & o & o & x \\ x & x & x & x & x & x \end{pmatrix}, \qquad p' = \begin{pmatrix} A & A & A & x & x & x \\ A & A & A & o & o & x \\ A & A & A & o & o & x \\ A & A & A & x & x & x \end{pmatrix},$$

$$\mathcal{M} = \begin{array}{|c|c|c|c|c|c|} \hline x,1\boxtimes 1\boxplus(1,1) & x,1\boxtimes 1\boxplus(1,2) & x,1\boxtimes 1\boxplus(1,3) & x,1\boxtimes 1\boxplus(1,4) & x,1\boxtimes 1\boxplus(1,5) & x,1\boxtimes 1\boxplus(1,6) \\ A,4\boxtimes 3\boxplus(1,1) & A,4\boxtimes 3\boxplus(1,1) & A,4\boxtimes 3\boxplus(1,1) & & & \\ \hline x,1\boxtimes 1\boxplus(2,1) & o,1\boxtimes 1\boxplus(2,2) & o,1\boxtimes 1\boxplus(2,3) & o,1\boxtimes 1\boxplus(2,4) & o,1\boxtimes 1\boxplus(2,5) & x,1\boxtimes 1\boxplus(2,6) \\ A,4\boxtimes 3\boxplus(1,1) & A,4\boxtimes 3\boxplus(1,1) & A,4\boxtimes 3\boxplus(1,1) & & & \\ \hline x,1\boxtimes 1\boxplus(3,1) & o,1\boxtimes 1\boxplus(3,2) & o,1\boxtimes 1\boxplus(3,3) & o,1\boxtimes 1\boxplus(3,4) & o,1\boxtimes 1\boxplus(3,5) & x,1\boxtimes 1\boxplus(3,6) \\ A,4\boxtimes 3\boxplus(1,1) & A,4\boxtimes 3\boxplus(1,1) & A,4\boxtimes 3\boxplus(1,1) & & & \\ \hline x,1\boxtimes 1\boxplus(4,1) & x,1\boxtimes 1\boxplus(4,2) & x,1\boxtimes 1\boxplus(4,3) & x,1\boxtimes 1\boxplus(4,4) & x,1\boxtimes 1\boxplus(4,5) & x,1\boxtimes 1\boxplus(4,6) \\ A,4\boxtimes 3\boxplus(1,1) & A,4\boxtimes 3\boxplus(1,1) & A,4\boxtimes 3\boxplus(1,1) & & & \\ \hline \end{array}$$

We now introduce the concept of tile compatibility, which is crucial for our algorithm. It is used to check if a tile used in a picture is compatible with tiles appearing in a rule, and its possible application area.

Definition 21. Compatibility. *Consider a tile t and a sub-multipicture $\mathcal{M}$ of the same size (m,n) over the same alphabet Σ. The* compatibility *between $\mathcal{M}$ and t, denoted as $\mathcal{M} :: t$, is a set of monopictures of size (m,n). Let each cell of such a monopicture be $M_{i,j} = (u_{i,j}, \alpha_{i,j})$.* $\quad M \in \mathcal{M} :: t \quad$ *iff* $\quad \forall (i,j),(k,l) \in m \boxtimes n$

$$u_{i,j} = t_{i,j} \ \wedge \ M_{i,j} \in \mathcal{M}_{i,j} \ \wedge \ (\alpha_{i,j} = \alpha_{k,l} \wedge u_{i,j} = u_{k,l} \ \vee \ \alpha_{i,j} \cap \alpha_{k,l} = \emptyset).$$

We say that $\mathcal{M}$ is compatible *with t iff $\mathcal{M} :: t \neq \emptyset$.*

Informally, a monopicture $M \in \mathcal{M} :: t$ contains in each cell (i, j) exactly one pair (A, α), taken from the ones in cell (i, j) of $\mathcal{M}$, with the constraint that A is the same as $t_{i,j}$. For each pair of cells in M, scopes are either disjoint or identical; when they are identical, also the symbol is identical. These constraints enforce correct bottom-up recognition of application areas, as better explained in the next section.

Example 9. Consider p, p' and $\mathcal{M}$ from the previous example. We report the compatibilities between the sub-multipicture $\mathcal{M}[2 \boxtimes 2 \boxplus (1,1)]$ and two tiles:

$$\mathcal{M}[2 \boxtimes 2 \boxplus (1,1)] :: \begin{pmatrix} x & x \\ x & o \end{pmatrix} = \left\{ \begin{pmatrix} (x, 1 \boxtimes 1 \boxplus (1,1)) & (x, 1 \boxtimes 1 \boxplus (1,2)) \\ (x, 1 \boxtimes 1 \boxplus (2,1)) & (o, 1 \boxtimes 1 \boxplus (2,2)) \end{pmatrix} \right\}$$

$$\mathcal{M}[2 \boxtimes 2 \boxplus (1,1)] :: \begin{pmatrix} A & A \\ A & A \end{pmatrix} = \left\{ \begin{pmatrix} (A, 4 \boxtimes 3 \boxplus (1,1)) & (A, 4 \boxtimes 3 \boxplus (1,1)) \\ (A, 4 \boxtimes 3 \boxplus (1,1)) & (A, 4 \boxtimes 3 \boxplus (1,1)) \end{pmatrix} \right\}$$

3.2 An informal description of the algorithm

At a high level, our algorithm is a 2D version on the well-known CKY, as it works bottom-up, and recognizes subpictures as the result of application of grammar rules, starting from the simplest subpictures of the input, i.e. its pixels. The main step of the algorithm (Step 1) defines its core, and is divided in three substeps: the first one, called also "match", is used to match tiles on the picture. The second substep, called also "grow", is used to find maximal areas compatible with a given rule. The last substep, called also "recognize", checks if there exist areas in which all the tiles of a rule are used. Those are therefore valid application areas, and their recognition is duly recorded. The algorithm succeeds only if it finds an application of a starting rule, having an application area which spans the whole picture.

Let us now consider the details of the steps. The initialization phase sets the tableau empty, then creates the multipicture from the input picture: if symbol x appears in cell $p_{i,j}$, then it puts pair $(x, 1 \boxtimes 1 \boxplus (i, j))$ in $M_{i,j}$.

In step 1.1 ("Match"), we scan each 2×2 subpicture t of the multipicture. That subpicture may appear among the tiles ω in the right-hand side of one or more rules R_e. Whenever this happens, we say that the subpicture is *tileable* by rule R_e. We locate the tableau elementary cell corresponding to the coordinates of that subpicture, and there we add a candidate (R_e, ω_x, α); ω_x contains all the tiles in ω except for t; α is the ceiling of the scopes of all used symbols in t. At the end of this step, we know whether each 2×2 subpicture is tileable by some rule.

In step 1.2 ("Grow"), we determine the same tileability information for all larger subpictures. We do so by merging candidates already in $T_{2,2}$, without examining the multipicture again. For example, if two 2×2 subpictures at (i, j) and $(i, j + 1)$ are tileable with rule R_e, then also the 2×3 subpicture at (i, j), which includes both the above subpictures, is tileable with the same rule. The set of missing tiles for the new subpicture is the intersection of the respective sets, while the scope of the new subpicture is the ceiling of the respective

scopes. Operatively, this means that when we find the candidate $(R_e, \omega_1, \alpha_1, s_1)$ in elementary cell $T[2 \boxtimes 2 \boxplus (i,j)]$ and the candidate $(R_e, \omega_2, \alpha_2, s_2)$ in adjacent elementary cell $T[2 \boxtimes 2 \boxplus (i,j+1)]$, a new candidate could be added to $T[2 \boxtimes 3 \boxplus (i,j)]$. We add this new candidate only if subpictures s_1 and s_2 overlap correctly horizontally, i.e. all the columns in s_1 except the first one are respectively equal to the columns in s_2 except the last one, i.e. $s_1 \parallel s_2$ is defined. The new candidate is $(R_e, \omega_1 \cap \omega_2, \lceil \alpha_1 \cup \alpha_2 \rceil, s_1 \parallel s_2)$. We proceed iteratively for all larger subpictures.

In step 1.3 ("Recognize"), we scan the tableau for complete candidates, i.e., candidates with no missing tiles. When we find such an entry $(R_e, \emptyset, \alpha, s)$ in $T[r \boxtimes c \boxplus (i,j)]$, it means that subpicture of size (r,c) at position (i,j) is tileable with rule $R_e = A \rightarrow \omega$, and additionally, all the tiles ω are used at least once in the subpictures. Therefore, we declare that subpicture $p[r \boxtimes c \boxplus (i,j)]$ belongs to $LOC(\omega)$, therefore there exists a derivation from an appropriate A-homogeneous subpicture to the current subpicture. This means that cells of coordinates $r \boxtimes c \boxplus (i,j)$ are generable by non-terminal symbol A, so add an entry $(A, r \boxtimes c \boxplus (i,j))$ to all those coordinates in the multipicture.

At the end of step 1.3, we may or not have added new entries to the multipicture. If we did, the multipicture contains new entries, that may render new subpictures tileable in other ways, so we repeat steps 1.1, 1.2 and 1.3 again. If we did not, then no more subpictures can be recognized, because we reached a fixed point.

The final step determines if each cell of the multipicture contains an $(S, m \boxtimes n \boxplus (1,1))$ entry or not[2]. In this case, we recognized the whole picture as a derivation of the starting symbol and we declare that picture p belongs to $\mathcal{L}(G)$; otherwise $p \notin \mathcal{L}(G)$.

3.3 The algorithm

For the sake of simplicity, we assume that all rules in R are variable-size rules, the extension of the algorithm to fixed-size rules is trivial. We also assume that rules are numbered as $R_1, R_2, ...$, their right-hand sides as $\omega_1, \omega_2, ...$, and the tiles appearing in ω_i as $t_{i,1}, t_{i,2}, ...$

Given a TRG grammar $G = (\Sigma, N, S, R)$ and a picture $p \in \Sigma^{(m,n)}$, the following algorithm determines whether $p \in \mathcal{L}(G)$ or not. The algorithm constructs and updates a tableau T of size (m,n) and a multipicture $\mathcal{M}$ of size (m,n) over alphabet $\Sigma \cup N$. The input grammar may be ambiguous.

- Step 0, "Initialize":
  Set the tableau empty: $\forall r, c, i, j : \; T[r \boxtimes c \boxplus (i,j)] = \emptyset$.
  Initialize the multipicture $\mathcal{M}$ as follows: $\mathcal{M}_{i,j} = \{(p_{i,j}, 1 \boxtimes 1 \boxplus (i,j))\}$.

- Step 1: repeat the following steps until fixed point is reached;

[2] By construction of step 1.3, if a cell contains an entry $(A, r \boxtimes c \boxplus (i,j))$, then all the cells in $r \boxtimes c \boxplus (i,j)$ also contain that entry. Therefore, if any cell contains $(S, m \boxtimes n \boxplus (1,1))$, than all the cells contain it.

- Step 1.1, "Match":
  if $\exists R_e, t, M$ such that
  $$R_e = (A \to \omega) \in R, \quad t \in \omega, \quad M \in (\mathcal{M}[2 \boxtimes 2 \boxplus (i,j)] :: t),$$
  where $M = \begin{pmatrix} (t_{1,1}, \alpha_{1,1}) & (t_{1,2}, \alpha_{1,2}) \\ (t_{2,1}, \alpha_{2,1}) & (t_{2,2}, \alpha_{2,2}) \end{pmatrix}$,
  then add $(R_e, \omega - t, \lceil \alpha_{1,1} \cup \alpha_{1,2} \cup \alpha_{2,1} \cup \alpha_{2,2} \rceil, t)$ to $T[2 \boxtimes 2 \boxplus (i,j)]$.

- Step 1.2, "Grow":
  $\forall r, c, 2 < r \leq m, 2 < c \leq n$ in lexicographical order:
  - if $(R_e, \omega_1, \alpha_1, s_1) \in T[r \boxtimes (c-1) \boxplus (i,j)]$,
    and $(R_e, \omega_2, \alpha_2, s_2) \in T[r \boxtimes (c-1) \boxplus (i,j+1)]$,
    and $s_1 \parallel s_2$ is defined,
    then add $(R_e, \omega_1 \cap \omega_2, \lceil \alpha_1 \cup \alpha_2 \rceil, s_1 \parallel s_2)$ to $T[r \boxtimes c \boxplus (i,j)]$;
  - if $(R_e, \omega_1, \alpha_1, s_1) \in T[(r-1) \boxtimes c \boxplus (i,j)]$,
    and $(R_e, \omega_2, \alpha_2, s_2) \in T[(r-1) \boxtimes c \boxplus (i+1,j)]$,
    and $\frac{s_1}{\overline{s_2}}$ is defined,
    then add $(R_e, \omega_1 \cap \omega_2, \lceil \alpha_1 \cup \alpha_2 \rceil, \frac{s_1}{\overline{s_2}})$ to $T[r \boxtimes c \boxplus (i,j)]$.

- Step 1.3, "Recognize":
  if $(R_e, \emptyset, \alpha, s) \in T[r \boxtimes c \boxplus (i,j)]$ and $\alpha \subseteq r \boxtimes c \boxplus (i,j)$ and $R_e = (A \to ...)$
  then $\forall (i', j') \in r \boxtimes c \boxplus (i,j)$ add $(A, r \boxtimes c \boxplus (i,j))$ to $\mathcal{M}_{i',j'}$.

- Step 2, "Declare":
  if $\forall (i,j) \in r \boxtimes c \ (S, m \boxtimes n \boxplus (1,1)) \in \mathcal{M}_{i,j}$
  then declare $p \in \mathcal{L}(G)$
  else declare $p \notin \mathcal{L}(G)$.

4 Conclusions

We presented a general bottom-up algorithm to parse languages generated by TRGs. This opens interesting possibilities of application of the presented description techniques, e.g. pattern recognition and image compression, still to be investigated. From [5, 6] we know that the class of TRG languages strictly contains REC. This, together with the constructive proofs there presented, make it straightforward to adapt and simplify the algorithm also for parsing REC languages.

It is worth mentioning that we implemented a fully functioning parser prototype. The prototype is written in Tcl [11], and we used it to generate the full example reported in the appendix.

References

1. 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.
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. Katsushi Inoue and Akira Nakamura. Some properties of two-dimensional on-line tessellation acceptors. *Information Sciences*, 13:95–121, 1977.
4. 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.
5. 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.
6. Stefano Crespi Reghizzi and Matteo Pradella. Tile Rewriting Grammars and Picture Languages. *Theoretical Computer Science*, 2005 (to appear), to download a draft: http://www.elet.polimi.it/upload/pradella/papers/trgTCS_rev.pdf.
7. Gennaro Costagliola, Andrea De Lucia, Sergio Orefice, and Genoveffa Tortora. A parsing methodology for the implementation of visual systems. *IEEE Trans. Softw. Eng.*, 23(12):777–799, 1997.
8. Azriel Rosenfeld. *Picture Languages (Formal Models for Picture Recognition)*. Academic Press, 1979.
9. Andrew Mercer and Azriel Rosenfeld. An array grammar programming system. *Communications of the ACM*, 16(5):299–305, 1973.
10. D. H. Younger. Recognition of context-free languages in time n^3. *Information and Control*, 10(2):189–208, 1967.
11. J. K. Ousterhout. *Tcl and the Tk Toolkit*. Addison-Wesley, 1994.

Appendix: A full example

In this section we illustrate how our parsing technique operates, by showing a complete example. More precisely, we illustrate how it recognizes picture p according to grammar G, where p and G are the same as in Example 6 at page 6. For ease of reference, we now assign a name to each tile in a right-hand side, as follows:

$$R_1 : S \to \left[\!\!\left[\begin{matrix} A\ A\ B\ B \\ A\ A\ B\ B \end{matrix}\right]\!\!\right] = \left\{ t_{1,1} = \begin{pmatrix} A\ A \\ A\ A \end{pmatrix}, t_{1,2} = \begin{pmatrix} A\ B \\ A\ B \end{pmatrix}, t_{1,3} = \begin{pmatrix} B\ B \\ B\ B \end{pmatrix} \right\}$$

$$R_2 : A \to \left[\!\!\left[\begin{matrix} x\ x\ x \\ x\ o\ o \\ x\ o\ o \\ x\ x\ x \end{matrix}\right]\!\!\right] = \left\{ t_{2,1} = \begin{pmatrix} x\ x \\ x\ o \end{pmatrix}, t_{2,2} = \begin{pmatrix} x\ x \\ o\ o \end{pmatrix}, t_{2,3} = \begin{pmatrix} x\ o \\ x\ o \end{pmatrix}, t_{2,4} = \begin{pmatrix} x\ o \\ x\ x \end{pmatrix}, t_{2,5} = \begin{pmatrix} o\ o \\ x\ x \end{pmatrix}, t_{2,6} = \begin{pmatrix} o\ o \\ o\ o \end{pmatrix} \right\}$$

$$R_3 : B \to \left[\!\!\left[\begin{matrix} x\ x\ x \\ o\ o\ x \\ o\ o\ x \\ x\ x\ x \end{matrix}\right]\!\!\right] = \left\{ t_{3,1} = \begin{pmatrix} x\ x \\ o\ x \end{pmatrix}, t_{3,2} = \begin{pmatrix} x\ x \\ o\ o \end{pmatrix}, t_{3,3} = \begin{pmatrix} o\ x \\ o\ x \end{pmatrix}, t_{3,4} = \begin{pmatrix} o\ x \\ x\ x \end{pmatrix}, t_{3,5} = \begin{pmatrix} o\ o \\ x\ x \end{pmatrix}, t_{3,6} = \begin{pmatrix} o\ o \\ o\ o \end{pmatrix} \right\}$$

As in Section 3.3, we denote the tableau and multipicture used by the algorithm with T and $\mathcal{M}$ respectively. Note that in step 1.1 the algorithm examines $\mathcal{M}$ and adds candidates to elementary cells in $T_{2,2}$; in step 1.2 it examines candidates from cell $T_{2,2}$ and adds candidates to other tableau cells; in step 1.3 it examines the entire T and adds elements to $\mathcal{M}$.

First, we show the initialization (step 0), then two iterations of step 1, and finally step 2. We omit a third iteration of step 1, which has no effects on T and M, because the fixed point is reached.

Step 0: initialize

At initialization time, T is set to empty. The multipicture $\mathcal{M}$ is initially set to the contents of the picture, with the *scope* of each terminal symbol set to the its coordinate in p.

$\mathcal{M} =$

$(x, 1\boxtimes 1\boxplus(1,1))$	$(x, 1\boxtimes 1\boxplus(1,2))$	$(x, 1\boxtimes 1\boxplus(1,3))$	$(x, 1\boxtimes 1\boxplus(1,4))$	$(x, 1\boxtimes 1\boxplus(1,5))$	$(x, 1\boxtimes 1\boxplus(1,6))$
$(x, 1\boxtimes 1\boxplus(2,1))$	$(o, 1\boxtimes 1\boxplus(2,2))$	$(o, 1\boxtimes 1\boxplus(2,3))$	$(o, 1\boxtimes 1\boxplus(2,4))$	$(o, 1\boxtimes 1\boxplus(2,5))$	$(x, 1\boxtimes 1\boxplus(2,6))$
$(x, 1\boxtimes 1\boxplus(3,1))$	$(o, 1\boxtimes 1\boxplus(3,2))$	$(o, 1\boxtimes 1\boxplus(3,3))$	$(o, 1\boxtimes 1\boxplus(3,4))$	$(o, 1\boxtimes 1\boxplus(3,5))$	$(x, 1\boxtimes 1\boxplus(3,6))$
$(x, 1\boxtimes 1\boxplus(4,1))$	$(o, 1\boxtimes 1\boxplus(4,2))$	$(o, 1\boxtimes 1\boxplus(4,3))$	$(o, 1\boxtimes 1\boxplus(4,4))$	$(o, 1\boxtimes 1\boxplus(4,5))$	$(x, 1\boxtimes 1\boxplus(4,6))$
$(x, 1\boxtimes 1\boxplus(5,1))$	$(x, 1\boxtimes 1\boxplus(5,2))$	$(x, 1\boxtimes 1\boxplus(5,3))$	$(x, 1\boxtimes 1\boxplus(5,4))$	$(x, 1\boxtimes 1\boxplus(5,5))$	$(x, 1\boxtimes 1\boxplus(5,6))$

Step 1.1 "match" (first iteration)

Tableau cells corresponding to rectangles of size $(1, c)$ and $(r, 1)$ are not used by the algorithm, and remain empty. We add elements to elementary cells $T[2 \boxtimes$

$2 \boxplus (i,j)]$ in the tableau: for each tile t of size $(2,2)$ in the right-hand size of a rule R_e, which is compatible with a $(2,2)$ sub-multipicture, we add a $(R_e, \omega_e - t, 2 \boxtimes 2 \boxplus (i,j))$ entry. For elementary cell $T[2 \boxtimes 2 \boxplus (1,1)]$:

$$\mathcal{M} = \begin{array}{|c|c|c|}\hline (\mathbf{x}, 1\boxtimes 1 \boxplus (\mathbf{1},\mathbf{1})) & (\mathbf{x}, 1\boxtimes 1 \boxplus (\mathbf{1},\mathbf{2})) & \ldots \\ \hline (\mathbf{x}, 1\boxtimes 1 \boxplus (\mathbf{2},\mathbf{1})) & (\mathbf{o}, 1\boxtimes 1 \boxplus (\mathbf{2},\mathbf{2})) & \ldots \\ \hline \ldots & \ldots & \ldots \\ \hline \end{array}$$

$$\Downarrow$$

$$T_{2,2} = \begin{array}{|c|c|}\hline (\mathbf{R_2}, \{\mathbf{t_{2,2}}, \mathbf{t_{2,3}}, \mathbf{t_{2,4}}, \mathbf{t_{2,5}}, \mathbf{t_{2,6}}\}, \mathbf{2} \boxtimes \mathbf{2} \boxplus (\mathbf{1},\mathbf{1})) & \ldots \\ \hline \ldots & \ldots \\ \hline \end{array}$$

The sub-multipicture tile corresponding to elementary cell $T[2 \boxtimes 2 \boxplus (1,2)]$ appears in the right-hand side of two rules (R_2 and R_3), therefore we add two candidates to $T[2 \boxtimes 2 \boxplus (1,2)]$:

$$\mathcal{M} = \begin{array}{|c|c|c|c|}\hline (x, 1\boxtimes 1 \boxplus (1,1)) & (\mathbf{x}, 1\boxtimes 1 \boxplus (\mathbf{1},\mathbf{2})) & (\mathbf{x}, 1\boxtimes 1 \boxplus (\mathbf{1},\mathbf{3})) & \ldots \\ \hline (x, 1\boxtimes 1 \boxplus (2,1)) & (\mathbf{o}, 1\boxtimes 1 \boxplus (\mathbf{2},\mathbf{2})) & (\mathbf{o}, 1\boxtimes 1 \boxplus (\mathbf{2},\mathbf{3})) & \ldots \\ \hline \ldots & \ldots & \ldots & \\ \hline \end{array}$$

$$\Downarrow$$

$$T_{2,2} = \begin{array}{|c|c|c|}\hline (R_2, \{t_{2,2}, t_{2,3}, t_{2,4}, t_{2,5}, t_{2,6}\}, 2 \boxtimes 2 \boxplus (1,1)) & (\mathbf{R_2}, \{\mathbf{t_{2,1}}, \mathbf{t_{2,3}}, \mathbf{t_{2,4}}, \mathbf{t_{2,5}}, \mathbf{t_{2,6}}\}, \mathbf{2} \boxtimes \mathbf{2} \boxplus (\mathbf{1},\mathbf{2})) & \ldots \\ & (\mathbf{R_3}, \{\mathbf{t_{3,1}}, \mathbf{t_{3,3}}, \mathbf{t_{3,4}}, \mathbf{t_{3,5}}, \mathbf{t_{3,6}}\}, \mathbf{2} \boxtimes \mathbf{2} \boxplus (\mathbf{1},\mathbf{2})) & \ldots \\ \hline \ldots & \ldots & \\ \hline \end{array}$$

Step 1.2 "grow" (first iteration)

For all the tableau cells from $T_{2,3}$ to $T_{2,6}$, we fill each elementary cell by "merging" the contents of a pair of elementary cells belonging to the tableau cell immediately on the left:

$$\text{If } \left.\begin{array}{l}(R_e, \omega_1, \alpha_1) \in T[r \boxtimes (c-1) \boxplus (i,j)] \wedge \\ (R_e, \omega_2, \alpha_2) \in T[r \boxtimes (c-1) \boxplus (i,j+1)]\end{array}\right\} \text{ then } (R_e, \omega_1 \cap \omega_2, \lceil \alpha_1 \cup \alpha_2 \rceil) \in T[r \boxtimes c \boxplus (i,j)].$$

$$T_{2,2} = \begin{array}{|c|c|c|}\hline (\mathbf{R_2}, \{\mathbf{t_{2,2}}, \mathbf{t_{2,3}}, \mathbf{t_{2,4}}, \mathbf{t_{2,5}}, \mathbf{t_{2,6}}\}, \mathbf{2} \boxtimes \mathbf{2} \boxplus (\mathbf{1},\mathbf{1})) & (\mathbf{R_2}, \{\mathbf{t_{2,1}}, \mathbf{t_{2,3}}, \mathbf{t_{2,4}}, \mathbf{t_{2,5}}, \mathbf{t_{2,6}}\}, \mathbf{2} \boxtimes \mathbf{2} \boxplus (\mathbf{1},\mathbf{2})) & \ldots \\ & (R_3, \{t_{3,1}, t_{3,3}, t_{3,4}, t_{3,5}, t_{3,6}\}, 2 \boxtimes 2 \boxplus (1,2)) & \ldots \\ \hline \ldots & \ldots & \\ \hline \end{array}$$

$$\Downarrow$$

$$T_{2,3} = \begin{array}{|c|c|}\hline (\mathbf{R_2}, \{\mathbf{t_{2,3}}, \mathbf{t_{2,4}}, \mathbf{t_{2,5}}, \mathbf{t_{2,6}}\}, \mathbf{2} \boxtimes \mathbf{3} \boxplus (\mathbf{1},\mathbf{1})) & \ldots \\ \hline \ldots & \ldots \\ \hline \end{array}$$

Then, we fill elementary cells in $T_{3,2}$ starting from elementary cells in $T_{2,2}$:

$$\text{If } \left.\begin{array}{l}(R_e, \omega_1, \alpha_1) \in T[(r-1) \boxtimes c \boxplus (i,j)] \quad \wedge \\ (R_e, \omega_2, \alpha_2) \in T[(r-1) \boxtimes c \boxplus (i+1,j)]\end{array}\right\} \text{ then } (R_e, \omega_1 \cap \omega_2, \lceil \alpha_1 \cup \alpha_2 \rceil) \in T[r \boxtimes c \boxplus (i,j)].$$

$T_{2,2} =$

$(\mathbf{R_2}, \{\mathbf{t_{2,2}}, \mathbf{t_{2,3}}, \mathbf{t_{2,4}}, \mathbf{t_{2,5}}, \mathbf{t_{2,6}}\}, \mathbf{2 \boxtimes 2 \boxplus (1,1)})$	$(R_2, \{t_{2,1}, t_{2,3}, t_{2,4}, t_{2,5}, t_{2,6}\}, 2 \boxtimes 2 \boxplus (1,2))$ ... $(R_3, \{t_{3,1}, t_{3,3}, t_{3,4}, t_{3,5}, t_{3,6}\}, 2 \boxtimes 2 \boxplus (1,2))$ ...
$(\mathbf{R_2}, \{\mathbf{t_{2,1}}, \mathbf{t_{2,2}}, \mathbf{t_{2,4}}, \mathbf{t_{2,5}}, \mathbf{t_{2,6}}\}, \mathbf{2 \boxtimes 2 \boxplus (2,1)})$	...
...	...

$\Downarrow\Downarrow$

$T_{3,2} =$

$(\mathbf{R_2}, \{\mathbf{t_{2,2}}, \mathbf{t_{2,4}}, \mathbf{t_{2,5}}, \mathbf{t_{2,6}}\}, \mathbf{3 \boxtimes 2 \boxplus (1,1)})$	...
...	...

Finally, we determine the contents of the remaining cells in the tableau by applying the same horizontal and vertical merging rules as above.

Step 1.3 "recognize" (first iteration)

Let us consider the final state of tableau cell $T_{5,3}$: it contains complete candidates for rules R_2 and R_3.

$(\mathbf{R2}, \emptyset, \mathbf{5 \boxtimes 3 \boxplus (1,1)})$	$(R2, \{t_{2,1}, t_{2,3}, t_{2,4}\} 5 \boxtimes 3 \boxplus (1,2))$ $(R3, \{t_{3,1}, t_{3,3}, t_{3,4}\}, 5 \boxtimes 3 \boxplus (1,2))$	$(R2, \{t_{2,1}, t_{2,3}, t_{2,4}\}, 5 \boxtimes 3 \boxplus (1,3))$ $(R3, \{t_{3,1}, t_{3,3}, t_{3,4}\}, 5 \boxtimes 3 \boxplus (1,3))$	$(\mathbf{R3}, \emptyset, \mathbf{5 \boxtimes 3 \boxplus (1,4)})$

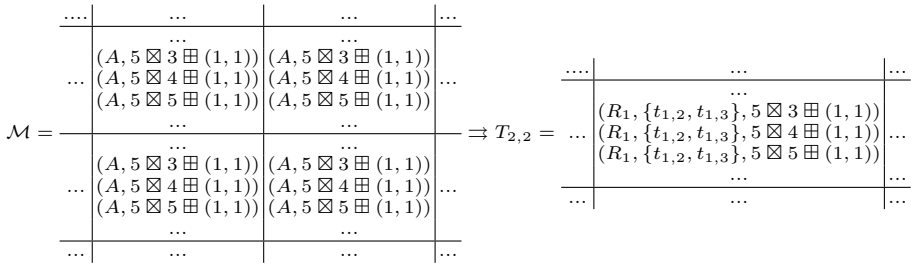
Therefore we add a pair $(A, 5 \boxtimes 3 \boxplus (1,1))$ to all cells of coordinates $5 \boxtimes 3 \boxplus (1,1)$ in $\mathcal{M}$, and a pair $(B, 5 \boxtimes 3 \boxplus (1,4))$ to all cells of coordinates $5 \boxtimes 3 \boxplus (1,4)$ in $\mathcal{M}$. The final state of the multipicture at the end of iteration 1 is given below.

$\mathcal{M} =$

$(x, 1 \boxtimes 1 \boxplus (1,1))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$	$(x, 1 \boxtimes 1 \boxplus (1,2))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$	$(x, 1 \boxtimes 1 \boxplus (1,3))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,1))$	$(x, 1 \boxtimes 1 \boxplus (1,4))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$	$(x, 1 \boxtimes 1 \boxplus (1,5))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$	$(x, 1 \boxtimes 1 \boxplus (1,6))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$
$(x, 1 \boxtimes 1 \boxplus (2,1))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$	$(o, 1 \boxtimes 1 \boxplus (2,2))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$	$(o, 1 \boxtimes 1 \boxplus (2,3))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,1))$	$(o, 1 \boxtimes 1 \boxplus (2,5))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$	$(o, 1 \boxtimes 1 \boxplus (2,5))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$	$(x, 1 \boxtimes 1 \boxplus (2,6))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$
$(x, 1 \boxtimes 1 \boxplus (3,1))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$	$(o, 1 \boxtimes 1 \boxplus (3,2))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$	$(o, 1 \boxtimes 1 \boxplus (3,3))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,1))$	$(o, 1 \boxtimes 1 \boxplus (3,4))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$	$(o, 1 \boxtimes 1 \boxplus (3,5))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$	$(x, 1 \boxtimes 1 \boxplus (3,6))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$
$(x, 1 \boxtimes 1 \boxplus (4,1))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$	$(o, 1 \boxtimes 1 \boxplus (4,2))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$	$(o, 1 \boxtimes 1 \boxplus (4,3))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,1))$	$(o, 1 \boxtimes 1 \boxplus (4,4))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$	$(o, 1 \boxtimes 1 \boxplus (4,5))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$	$(x, 1 \boxtimes 1 \boxplus (4,6))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$
$(x, 1 \boxtimes 1 \boxplus (5,1))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$	$(x, 1 \boxtimes 1 \boxplus (5,2))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$	$(x, 1 \boxtimes 1 \boxplus (5,3))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,1))$	$(x, 1 \boxtimes 1 \boxplus (5,4))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$	$(x, 1 \boxtimes 1 \boxplus (5,5))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$	$(x, 1 \boxtimes 1 \boxplus (5,6))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$

Step 1.1 "match" (second iteration)

We apply now Step 1.1 to the new multipicture. We match rule R_1 over entire A- and B-subpictures.



Step 1.2 "grow" (second iteration)

At the end of Step 1.2, cell $T_{5,6}$ contains the following complete candidate:

$$T_{2,2} = \boxed{(R_1, \emptyset, 5 \boxtimes 6 \boxplus (1,1))}\,.$$

Thus, an application area for rule R_1 was recognized over the entire multipicture.

Step 1.3 "recognize" (second iteration)

We add a pair $(S, 5 \boxtimes 6 \boxplus (1,1))$ to every cell in $\mathcal{M}$ in rectangle $5 \boxtimes 6 \boxplus (1,1)$ (representing the whole picture). The final state of $\mathcal{M}$ is:

$\mathcal{M} =$

$(x, 1 \boxtimes 1 \boxplus (1,1))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(x, 1 \boxtimes 1 \boxplus (1,2))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(x, 1 \boxtimes 1 \boxplus (1,3))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,1))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(x, 1 \boxtimes 1 \boxplus (1,4))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(x, 1 \boxtimes 1 \boxplus (1,5))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(x, 1 \boxtimes 1 \boxplus (1,6))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$
$(x, 1 \boxtimes 1 \boxplus (2,1))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(o, 1 \boxtimes 1 \boxplus (2,2))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(o, 1 \boxtimes 1 \boxplus (2,3))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,1))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(o, 1 \boxtimes 1 \boxplus (2,5))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(o, 1 \boxtimes 1 \boxplus (2,5))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(x, 1 \boxtimes 1 \boxplus (2,6))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$
$(x, 1 \boxtimes 1 \boxplus (3,1))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(o, 1 \boxtimes 1 \boxplus (3,2))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(o, 1 \boxtimes 1 \boxplus (3,3))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,1))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(o, 1 \boxtimes 1 \boxplus (3,4))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(o, 1 \boxtimes 1 \boxplus (3,5))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(x, 1 \boxtimes 1 \boxplus (3,6))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$
$(x, 1 \boxtimes 1 \boxplus (4,1))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(o, 1 \boxtimes 1 \boxplus (4,2))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(o, 1 \boxtimes 1 \boxplus (4,3))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,1))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(o, 1 \boxtimes 1 \boxplus (4,4))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(o, 1 \boxtimes 1 \boxplus (4,5))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(x, 1 \boxtimes 1 \boxplus (4,6))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$
$(x, 1 \boxtimes 1 \boxplus (5,1))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(x, 1 \boxtimes 1 \boxplus (5,2))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(x, 1 \boxtimes 1 \boxplus (5,3))$ $(A, 5 \boxtimes 3 \boxplus (1,1))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,1))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(x, 1 \boxtimes 1 \boxplus (5,4))$ $(A, 5 \boxtimes 4 \boxplus (1,1))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(x, 1 \boxtimes 1 \boxplus (5,5))$ $(A, 5 \boxtimes 5 \boxplus (1,1))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$	$(x, 1 \boxtimes 1 \boxplus (5,6))$ $(B, 5 \boxtimes 3 \boxplus (1,4))$ $(B, 5 \boxtimes 4 \boxplus (1,3))$ $(B, 5 \boxtimes 5 \boxplus (1,2))$ $(S, 5 \boxtimes 6 \boxplus (1,1))$

Step 2 "declare"

Each cell in the multipicture contains an entry $(S, 5 \boxtimes 6 \boxplus (1,1))$. The picture is therefore successfully recognized.