A CKY parser for picture grammars

Stefano Crespi Reghizzi, Matteo Pradella [*]

DEI – Politecnico di Milano and CNR IEIIT–MI
Piazza Leonardo da Vinci, 32,
I-20133 Milano, Italy

Abstract

We study the complexity of the membership or parsing problem for pictures generated by a family of picture grammars: Siromoney's Context-Free Kolam Array grammars (coincident with Matz's context-free picture grammars). We describe a new parsing algorithm, which extends the Cocke, Kasami and Younger's classical parsing technique for string languages and preserves the polynomial time complexity.

Key words: formal languages, picture languages, 2D languages, context-free picture grammars, CKY parsing, Kolam Array grammars, Matrix grammars, Siromoney grammars

1 Introduction

In pattern recognition and related areas several formal language theoretical approaches have been conceived to define and analyze pictures. A picture is a rectangular array of symbols of a finite alphabet, and a set of pictures constitutes a 2D or picture language. We are interested in the membership problem for certain formal models, which extend to 2D the basic families of string languages, namely the regular (REG) and context-free (CF). For all such models membership of a picture in a language is decidable, but the algorithmic complexities (with respect to the number of pixels) greatly differ. In fact some models have changed over time or have been reinvented with different names, so that terminology needs to be settled.

[⋆] This research has been supported in part by MUR grant PRIN 2005015419-002 "Linguaggi formali".

[*] Corresponding author.
Email addresses: `crespi@elet.polimi.it` (Stefano Crespi Reghizzi), `pradella@elet.polimi.it` (Matteo Pradella).

We mention that some grammar models are based on underlying string grammars, and their properties depend on the Chomsky type (context sensitive, CF, REG) of the latter. This is the case of Kolam grammars investigated here, for which we consider the CF sub-case.

Coming to membership problem (MP) complexity, some known results are listed next, for those formalisms which in different senses extend CF rules to two dimensions.

- For CF Isometric Array grammars [10] (equivalent to CF Puzzle grammars [8]), MP is NP-complete in time [7].
- For CF Matrix grammars [12], MP is polynomial-time [9]; this algorithm does not really differ from string parsing, since nonterminals generate strings (either vertical or horizontal ones).
- For Tiling Systems [2] (coincident with Wang Tiles [13]), MP is NP-complete in time [4], [5], hence also for Tile Rewriting grammars [1], which strictly include the former.

In this paper we address the problem of parsing pictures with Kolam or Matz grammars. To cope with subpictures, instead of substrings, our algorithm adds two dimensions to the Cocke, Kasami and Younger recognition matrix (CKY) [15]. It works bottom-up, and recognizes subpictures as a result of the application of grammar rules, starting from atomic subpictures, i.e. pixels. Adjacent parsed subpictures are then combined. We prove that the algorithm has polynomial time complexity. [1] The result applies of course to CF Matrix grammars too, since their translation to Kolam form is straightforward.

Since having a simple and efficient parser is certainly a prerequisite for any application to syntactic image processing, some applied research on pattern recognition has from time to time studied various picture grammar models, with rather ad hoc restrictions meant to reduce parsing complexity (see e.g. [14]).

To the best of our knowledge, this is the most general polynomial time parsing algorithm for picture grammars generalizing CF string grammars into two dimensions.

2 Pictures and Grammars

We briefly recall a few basic definitions, referring to [3] for more complete definitions. A *picture* on a finite alphabet Σ is a two-dimensional rectangular array of elements in Σ. The *size* $|p|$ of a picture p is the pair $(|p|_{row}, |p|_{col})$ of its number of rows and columns. A *pixel* $p(i,j)$, $1 \leq i \leq |p|_{row}, 1 \leq j \leq |p|_{col}$, is the element at

[1] An implementation is available here: http://www.elet.polimi.it/upload/pradella.

position (i,j) in the array p. The indices grow from top to bottom for the rows and from left to right for the columns. The empty picture, denoted λ, has size (0,0).
Let $\Sigma^{*,*}$ be the set of all pictures over Σ. A picture language over Σ is a subset of $\Sigma^{*,*}$.
Two picture-combining, partial operations are used: *column concatenation* is defined for all pictures p,q such that $|p|_{row} = |q|_{row}$, written $p \oplus q$, and represents the horizontal juxtaposition of p and q; *row concatenation* is defined for all pictures p,q such that $|p|_{col} = |q|_{col}$, written $p \ominus q$, and represents the vertical juxtaposition of p over q.
Let p,q be pictures. For every i,j, with $1 \leq i \leq |p|_{row}$, $1 \leq j \leq |p|_{col}$, q is a *subpicture* of p at position (i,j), if $1 \leq |q|_{row} \leq |p|_{row} - i + 1$, $1 \leq |q|_{col} \leq |p|_{col} - j + 1$, and $q(x,y) = p(i+x-1, j+y-1)$ for all $1 \leq x \leq |q|_{row}$, $1 \leq y \leq |q|_{col}$.
In the rest of the paper, we identify a subpicture q of a picture p by using the quadruple $(i,j;h,k)$, where (i,j) is the top-left coordinate of q in p, while (h,k) is the bottom-right coordinate. We will call such quadruples $\alpha, \alpha_1, \alpha_2, \ldots$. The notation $p_{(i,j;h,k)}$ will denote the subpicture q of p at position (i,j).

2.1 Context-Free Kolam Array grammars

This class of grammars has been introduced by Siromoney et al. [11] under the name "Array grammars", later renamed "Kolam Array grammars" in order to avoid confusion with Rosenfeld's homonymous model. Much later Matz reinvented the same model [6] (considering only CF rules).[2] We prefer to keep the historical name, CF Kolam grammars (CFKG), and to use the more succint definition of Matz.

Definition 1 *A* sentential form *over an alphabet V is a non-empty well-parenthesized expression using the two concatenation operators, $\ominus$ and $\oplus$, and symbols taken from V. $\mathcal{SF}(V)$ denotes the set of all sentential forms over V. A sentential form ϕ defines either one picture over V denoted by $[\![\phi]\!]$, or none.*

For example, $\phi_1 = ((a \oplus b) \ominus (b \oplus a)) \in \mathcal{SF}(\{a,b\})$ and $[\![\phi_1]\!]$ is the picture $\begin{matrix} a & b \\ b & a \end{matrix}$.
On the other hand $\phi_2 = ((a \oplus b) \ominus a)$ denotes no picture, since the two arguments of the $\ominus$ operator have different column numbers.

CF Kolam grammars are defined analogously to CF string grammars. Derivation is similar: a sentential form over terminal and nonterminal symbols results from the preceding one by replacing a non-terminal with some corresponding right hand side of a rule. The end of a derivation is reached when the sentential form does not

[2] We realized with some surprise that Matz's grammars coincide with Kolam grammars, but we believe the two models have never been compared before.

contain any nonterminal symbols. If this resulting form denotes a picture, then that picture is generated by the grammar.

Definition 2 *A* Context-Free Kolam grammar (CFKG) *is a tuple $G = (\Sigma, N, R, S)$, where Σ is the finite set of* terminal *symbols, disjoint from the set N of* non-terminal *symbols; $S \in N$ is the* start *symbol; and $R \subseteq N \times \mathcal{SF}(N \cup \Sigma)$ is the set of* rules. *A rule $(A, \phi) \in R$ will be written as $A \rightarrow \phi$.*

For a grammar G, we define the *derivation* relation $\Rightarrow_G$ on the sentential forms $\mathcal{SF}(N \cup \Sigma)$ by $\psi_1 \Rightarrow_G \psi_2$ iff there is some rule $A \rightarrow \phi$, such that ψ_2 results from ψ_1 by replacing an occurrence of A by $(\,\phi\,)$. As usual, $\overset{*}{\Rightarrow}_G$ denotes the reflexive and transitive closure. Notice that the derivation thus defined rewrites strings, not pictures.

From the derived sentential form, one then obtains the denoted picture. The picture language generated by G is the set

$$L(G) = \{[\![\psi]\!] \mid \psi \in \mathcal{SF}(\Sigma), S \overset{*}{\Rightarrow}_G \psi\}.$$

With a slight abuse of notation, we will often write $A \overset{*}{\Rightarrow}_G p$, with $A \in N, p \in \Sigma^{*,*}$, instead of $\exists \phi : A \overset{*}{\Rightarrow}_G \phi, [\![\phi]\!] = p$.

It is convenient to consider a normal form with exactly two or zero nonterminals in the right part of a rule [6].

Definition 3 *A grammar $G = (\Sigma, N, R, S)$, is in* Chomsky Normal Form *iff every rule in R has the form either $A \rightarrow t$, or $A \rightarrow B \ominus C$, or $A \rightarrow B \mathbin{\varobar} C$, where $A, B, C \in N$, and $t \in \Sigma$.*

We know from [6] that for every CFKG G, if $L(G)$ does not contain the empty picture, there exists a CFKG G' in Chomsky Normal Form, such that $L(G) = L(G')$. Also, the classical algorithm to translate a string grammar into Chomsky Normal Form can be easily adapted to CFKGs.

Example 1 *The following Chomsky Normal Form grammar G defines the set of pictures such that each column is a palindrome:*

$S \rightarrow V \mathbin{\varobar} S \mid A_1 \ominus A_2 \mid B_1 \ominus B_2 \mid a \mid b$
$V \rightarrow A_1 \ominus A_2 \mid B_1 \ominus B_2 \mid a \mid b$
$A_2 \rightarrow V \ominus A_1 \mid a$
$B_2 \rightarrow V \ominus B_1 \mid b$
$A_1 \rightarrow a$
$B_1 \rightarrow b.$

3 Parsing Context-Free Kolam Grammars

To present our version of the CKY algorithm, we have to generalize from substrings to subpictures. As a substring is identified by the positions of its first and last characters, a subpicture is conveniently identified by its top-left and bottom-right vertices.

Let p be an input picture, having size (m, n), to be parsed with a grammar $G = (\Sigma, N, R, S)$ in Chomsky Normal Form.

Definition 4 *A* recognition matrix $\mathfrak{M}$ *is a 4-dimensional* $m \times n \times m \times n$ *matrix, where* $\mathfrak{M}(i, j; h, k) \subseteq N$*, i.e. each element is a set of non-terminals. The meaning of* $A \in \mathfrak{M}(i, j; h, k)$ *is that* A *can derive the subpicture* $p_{(i,j;h,k)}$ *of* p*.*

In fact, only cells $(i, j; h, k)$, with $h \geq i, k \geq j$, are used: these cells are the four-dimensional counterpart of the upper triangular matrix used in classical CKY.

Informally, the parsing algorithm starts by considering terminal rules of the grammar (i.e. those like $A \to t$ of Definition 3), and places the corresponding non-terminal in the recognition matrix (Initialization phase). For instance, if $p(4, 5) = a$, and $A \to a$ is a rule, then A is placed in the set $\mathfrak{M}(4, 5; 4, 5)$.

The main part of the algorithm works in a bottom-up fashion, by considering subpictures covering larger and larger areas, and trying to decompose such areas either horizontally, or vertically, to match them with rules of the grammar.

For instance, consider the subpicture $q = p_{(1,2;3,4)}$.
Let us suppose that $B \in \mathfrak{M}(1, 2; 3, 2)$, $C \in \mathfrak{M}(1, 3; 3, 4)$, and $A \to B \oplus C$ is a grammar rule. Since $A \Rightarrow_G (B \oplus C) \overset{*}{\Rightarrow}_G (p_{(1,2;3,2)} \oplus p_{(1,3;3,4)})$, A is placed in the set $\mathfrak{M}(1, 2; 3, 4)$.

The algorithm terminates after considering the whole picture $p = p_{(1,1;m,n)}$: if $\mathfrak{M}(1, 1; m, n)$ contains S, then p is accepted.

Figure 1 contains the algorithm.

Example 2 *For the grammar of Example 1, we present the steps of the parsing of a sample picture (see Figure 2, left).*

Figure 2 represents the 4-dimensional matrix $\mathfrak{M}$ *as a directed labelled graph over a* $|p|_{row} \times |p|_{col}$ *grid, with sets of nonterminals as arc labels. An arc from node* (i, j) *to node* (i', j') *identifies subpicture* $p_{(i,j;i',j')}$*. The arc is labelled with a set that is the content of* $\mathfrak{M}(i, j; i', j')$*. Figure 2 presents* $\mathfrak{M}$ *after parsing* p*. Since* $|p| = (3, 2)$ *and* $S \in \mathfrak{M}(1, 1; 3, 2)$*, picture* p *is accepted.*

Theorem 1 $\mathfrak{M}(\alpha) = \{A \in N \mid A \overset{*}{\Rightarrow}_G p_\alpha\}$.

```
*Initialization*:
Every set in $\mathfrak{M}$ is empty;
**For each** pixel $p(i,j) = t$,
    **if** there exists a rule $A \to t \in R$,
    **then** put $A$ into the set $\mathfrak{M}(i,j;i,j)$.

*Main*:
**Remark**: the next two loops consider every size $(v,h) \in \{(1,1),\dots,(m,n)\}$.
**For each** $v$, with $1 \le v \le m$:
**For each** $h$, with $1 \le h \le n$:
    **For each** coordinate $(i,j) \in \{(1,1),\dots,(m,n)\}$:
    - let $\alpha = (i,j;i+v-1,j+h-1)$;
    - **Remark**: next consider every possible horizontal decomposition of $p_\alpha$ into $p_{\alpha_1} \oplus p_{\alpha_2}$.
      **For each** $k$, with $j < k < j+h-1$:
            let $\alpha_1 = (i,j;i+v-1,k), \alpha_2 = (i,k+1;i+v-1,j+h-1)$;
            **if** for some $A, B, C$,
            $B \in \mathfrak{M}(\alpha_1)$ and $C \in \mathfrak{M}(\alpha_2)$ and $A \to B \oplus C \in R$
            **then** put $A$ into the set $\mathfrak{M}(\alpha)$;
    - **Remark**: next consider every possible vertical decomposition of $p_\alpha$ into $p_{\alpha_1} \ominus p_{\alpha_2}$.
      **For each** $k$, with $i < k < i+v-1$:
            let $\alpha_1 = (i,j;k,j+h-1), \alpha_2 = (k+1,j;i+v-1,j+h-1)$;
            **if** for some $A, B, C$,
            $B \in \mathfrak{M}(\alpha_1)$ and $C \in \mathfrak{M}(\alpha_2)$ and $A \to B \ominus C \in R$
            **then** put $A$ into the set $\mathfrak{M}(\alpha)$.

*Acceptance*:
$p \in L(G) \iff S \in \mathfrak{M}(1,1;m,n)$.
```

Fig. 1. The parsing algorithm.

Proof The proof is by induction over derivation steps.

- *Base*: $\alpha = (i,j,i,j)$. This means that $|p_\alpha| = (1,1)$. Hence, $A \overset{*}{\Rightarrow}_G p_\alpha$ iff $A \to p_\alpha \in R$. This case is handled by the initialization procedure. If $p_\alpha = t$, and there exists a rule $A \to t$, then the algorithm puts A into $\mathfrak{M}(\alpha)$. Vice versa, $A \in \mathfrak{M}(\alpha)$ means that the algorithm has put A in the set, therefore there must exist a rule $A \to p_\alpha$.
- *Induction*: let us consider $\alpha = (i,j;i+v-1,j+h-1)$, $v > 1$, or $h > 1$, or both.
  - We prove that $A \overset{*}{\Rightarrow}_G p_\alpha$ implies $A \in \mathfrak{M}(\alpha)$.
    In this case, the size of the subpicture is not $(1,1)$, therefore the first rule used in the derivation $A \overset{*}{\Rightarrow}_G p_\alpha$ has either the form $A \to B \oplus C$, or the form $A \to B \ominus C$.
    First, consider the case $A \to B \oplus C$. This means that there must be a k, with $j < k < j+h-1$, $\alpha_1 = (i,j;i+v-1,k)$, $\alpha_2 = (i,k+1;i+v-1,j+h-1)$, such that $B \overset{*}{\Rightarrow}_G p_{\alpha_1}$, and $C \overset{*}{\Rightarrow}_G p_{\alpha_2}$. Thanks to the first two loops of the *Main* procedure, when the algorithm consider α, it has already considered both α_1

$$p = \begin{array}{|c|}\hline \text{a b} \\ \text{b b} \\ \text{a b} \\ \hline \end{array}$$

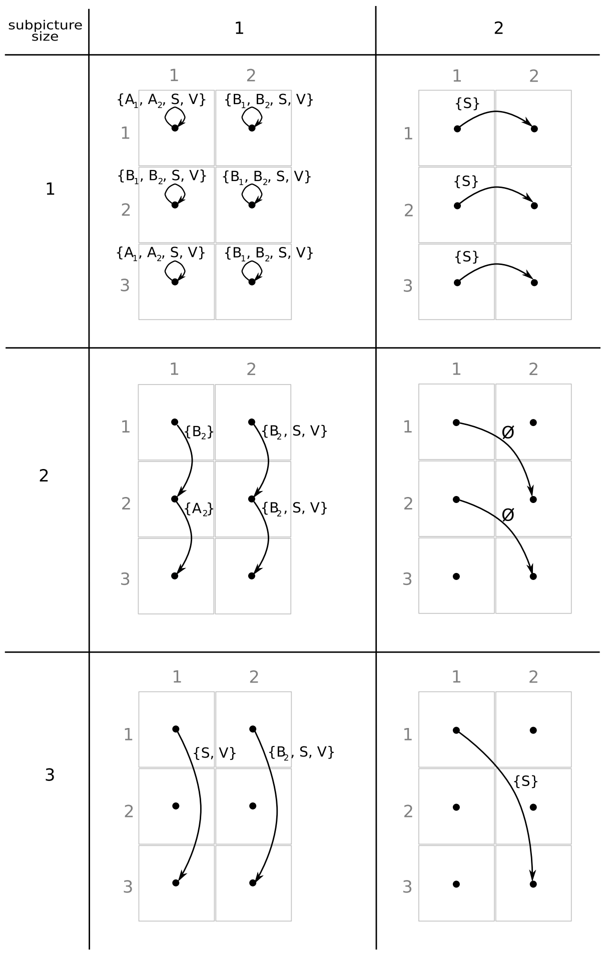

Fig. 2. Matrix $\mathfrak{M}$ after parsing picture p.

and α_2. By the induction hypothesis, $B \overset{*}{\Rightarrow}_G p_{\alpha_1}$ implies $B \in \mathfrak{M}(\alpha_1)$, and $C \overset{*}{\Rightarrow}_G p_{\alpha_2}$ implies $C \in \mathfrak{M}(\alpha_2)$. Hence, $B \in \mathfrak{M}(\alpha_1)$, $C \in \mathfrak{M}(\alpha_2)$. So, the algorithm puts A in $\mathfrak{M}(\alpha)$.
The case $A \to B \ominus C$ is analogous, with $i < k < i + v - 1$, $\alpha_1 = (i, j; k, j + h - 1)$, and $\alpha_2 = (k + 1, j; i + v - 1, j + h - 1)$.

- Next, we prove that $A \in \mathfrak{M}(\alpha)$ implies $A \overset{*}{\Rightarrow}_G p_\alpha$.
  $A \in \mathfrak{M}(\alpha)$ means that the *Main* procedure has put A in the set. Therefore, there must exist a rule having either the form $A \to B \oplus C$, or the form $A \to B \ominus C$. First, consider case $A \to B \oplus C$. There must exist a k, with $j < k < j+h-1$, $\alpha_1 = (i, j; i+v-1, k)$, $\alpha_2 = (i, k+1; i+v-1, j+h-1)$, such that $B \in \mathfrak{M}(\alpha_1)$, and $C \in \mathfrak{M}(\alpha_2)$. By the induction hypothesis, $B \in \mathfrak{M}(\alpha_1)$ implies $B \overset{*}{\Rightarrow}_G p_{\alpha_1}$, and $C \in \mathfrak{M}(\alpha_2)$ implies $C \overset{*}{\Rightarrow}_G p_{\alpha_2}$. Therefore, $B \overset{*}{\Rightarrow}_G p_{\alpha_1}$, and $C \overset{*}{\Rightarrow}_G p_{\alpha_2}$. But $A \to B \oplus C$ is a rule, hence $A \Rightarrow B \oplus C \overset{*}{\Rightarrow}_G p_{\alpha_1} \oplus p_{\alpha_2} = p_\alpha$.
  The other case, i.e. $A \to B \ominus C$, is analogous, with $i < k < i + v - 1$, $\alpha_1 = (i, j; k, j + h - 1)$, and $\alpha_2 = (k + 1, j; i + v - 1, j + h - 1)$. $\square$

Complexity: Let $|p| = (m, n)$. The initialization procedure is very simple and has complexity $m \cdot n$.

Let us now consider the main procedure. The first loop (control variable v) and second loop (control variable h) of the algorithm are on sets having size $m \cdot n$. The number of possible horizontal (resp. vertical) decompositions is always at most n (resp. m). Therefore, time complexity is $O\left((m \cdot n)^2(m+n)\right)$.
Space complexity: the size of $\mathfrak{M}$ is $\Theta\left((m \cdot n)^2\right)$.
Note that in the 1D case ($m = 1$) the complexity of the algorithm coincides with that of the classical CKY.

References

[1] S. Crespi Reghizzi and M. Pradella. Tile Rewriting Grammars and Picture Languages. *Theoretical Computer Science*, 340(2):257–272, 2005.

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

[3] D. Giammarresi and A. 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] H. Lewis. Complexity of solvable cases of the decision problem for predicate calculus. In *Proc. 19th Symposium on Foundations of Computer Science*, pages 35–47, 1978.

[5] K. Lindgren, C. Moore, and M. Nordahl. Complexity of two-dimensional patterns. *Journal of Statistical Physics*, 91(5-6):909–951, June 1998.

[6] O. 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.

[7] K. Morita, Y. Yamamoto, and K. Sugata. The complexity of some decision problems about two-dimensional array grammars. *Inf. Sci.*, 30(3):241–262, 1983.

[8] M. Nivat, A. Saoudi, K. G. Subramanian, R. Siromoney, and V. R. Dare. Puzzle grammars and context-free array grammars. *International Journal of Pattern Recognition and Artificial Intelligence*, 5:663–676, 1991.

[9] V. Radhakrishnan, V. T. Chakaravarthy, and K. Krithivasan. Pattern matching in matrix grammars. *J. Autom. Lang. Comb.*, 3(1):59–72, 1998.

[10] A. Rosenfeld. *Picture Languages (Formal Models for Picture Recognition)*. Academic Press, 1979.

[11] G. Siromoney, R. Siromoney, and K. Krithivasan. Picture languages with array rewriting rules. *Information and Control*, 23(5):447–470, 1973.

[12] R. Siromoney. On equal matrix languages. *Information and Control*, 14(2):135–151, February 1969.

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

[14] Y. Yamamoto and K. Morita. Two-dimensional uniquely parsable isometric array grammars. *Int. J. Pattern Recognition Artif. Intell.*, 6:301–313, 1992.

[15] D. H. Younger. Recognition of context-free languages in time n^3. *Information and Control*, 10(2):189–208, 1967.