ICDAR 2007: Proceedings of the 9th International Conference on Document Analysis and Recognition, Curitiba, Brazil, Sept 23-26, 2007, Volume 2, pp. 849-853, Publisher IEEE Computer Society, ISBN 0-7695-2822-8

Mathematical Formulae Recognition using 2D Grammars

Daniel Průša, Václav Hlaváč
Czech Technical University, Faculty of Electrical Engineering
Department for Cybernetics, Center for Machine Perception
121 35 Prague 2, Karlovo náměstí 13
{prusa,hlavac}@cmp.felk.cvut.cz, http://cmp.felk.cvut.cz

Abstract

We present a method for off-line mathematical formulae recognition based on the structural construction paradigm and two-dimensional grammars. In general, this approach can be successfully used in the analysis of images containing objects that exhibit rich structural relations. An important benefit of the structural construction is in treating the symbol segmentation in the image and its structural analysis as a single intertwined process. This allows the system to avoid errors usually appearing during the segmentation phase. We have developed and tested a pilot study proving that the method is computationally efficient, practical and able to cope with noise.

1 Introduction

Recognition of mathematical formulae has been a widely studied problem. Formulae contain significant structural information which is expressed as 2D spatial relations. Thus, mathematical formulae recognition is an appropriate application area for testing 2D grammars-based approach. The on-line formulae recognition is being used to process handwritten formulae entered trough tablets, while the off-line recognition serves to digitize scientific documents or notes. There are many contributions concerning this topic – several methods have been designed or adopted for the formulae recognition, a taxonomy can be found in [2].

Most of the known methods follow a two-phase procedure:

1. Detection of individual symbols by image segmentation and labeling symbols using pattern recognition techniques.

2. Structural analysis of relations among labeled symbols.

Our criticism of the commonly used methods concerns the image segmentation which is usually done without any knowledge of the formulae structure. It is hardly possible to recover from errors made during the symbol segmentation phase. There have been attempts to employ additional error corrections schemes. However, this postprocessing corrections do not fit naturally into the pattern recognition process.

There are works that partially deal with this problem. Local properties of mathematical structures are considered when computing the best segmentation for on-line formulae recognition in [11]. Furthermore, several best matches returned by an OCR tool for a segmented region are passed into the structural analysis in [5]. This allows to minimize a global penalty of recognition while possibly increasing a local penalty of recognized symbols. However, there are still many cases these two approaches cannot deal with.

Mathematical formulae exhibit a rich structure. We present a method based on structural construction paradigm [10] that benefits from this fact and tries to solve the stated problem by driving the segmentation by the structural analysis. Results achieved by implementing a pilot study aiming at the off-line recognition of mathematical formulae are summarized in this paper. We have already presented an introduction into 2D context-free (CF) grammars theory and results of the method for the off-line recognition of black and white, handwritten images of formulae in [8]. This contribution further extends ideas of the method, presents their application on gray-scale images and printed formulae, and gives more detailed experimental results.

The method itself is quite general and can be used for various types of structured images. The most similar approaches to our one have been already applied by others to the recognition of musical scores [9] or electrical circuits [4]. The applications of the method differ in formalisms used for the structural analysis. We have based it on two-dimensional grammars that were motivated by 2D CF grammars presented in [10]. The first author of this paper studied independently the theoretical limits of 2D CF grammars [7] and proved them to be rather restrictive. We use an extension of the grammars powerful enough to express the formulae structure. Furthermore, we focus on an

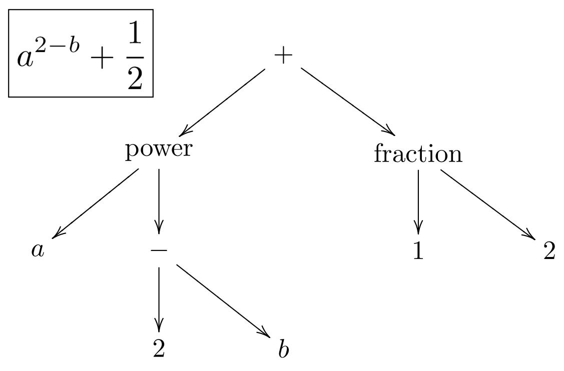

Figure 1. Input and output of recognition.

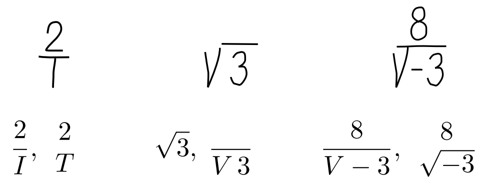
$\frac{2}{I}, \frac{2}{T}$
$\sqrt{3}, \overline{V3}$
$\frac{8}{V-3}, \frac{8}{\sqrt{-3}}$

Figure 2. Segmentation ambiguities.

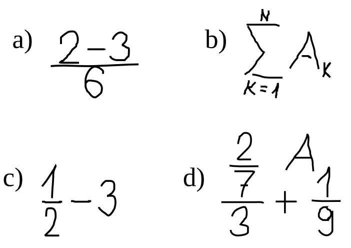

Figure 3. Cases we would like to handle.

efficient parsing algorithm for the grammars. Grammar-based formalisms for formulae description, some of them similar in some aspects to our grammars, appear often in the literature. They include geometric grammars [3] or graph grammars [5]. The notion of context-free grammars is presented in [6].

2 Formulae Recognition Task

The task of the formulae recognition is to produce a tree over elementary symbols that represents the formula structure in the input image. An example is shown in Figure 1.

We consider formulae consisting of common elements and constructs as numbers, variables, brackets, subscripts, superscripts, basic unary and binary operators, power to operations, fractions, sums, integrals and square roots.

We have required the proposed method to be able to deal with the following situations.

- Symbols touching vertically or horizontally.
- Symbols split into several components.
- Ambiguities.
- Misplaced symbols.
- Noise.

These cases can make the image segmentation hard, as it is demonstrated in Figure 2. There are two recognition results bellow each image – the first one obtained after the correct segmentation, while the second one after the incorrect segmentation (the segmented symbols does not form a valid formula in this case).

Figure 3 shows another examples causing difficulties for the recognition. Case a), resp. b) demonstrates a formula with touching, resp. split symbols. Case c) illustrates a fraction line and a minus sign represented by the same image, the meaning is being given by the context. And finally, case d) shows an additional symbol A included into the formula by a mistake. We require our method to exclude such a misplaced symbol and recognize the formula composed of the other symbols.

3 Applied Method

The main ideas taken from the structural construction, which we follow in our approach, can be expressed in the following way: Perform 'rough segmentation' of the input image. For each possible elementary symbol (*terminal*), find all occurrences of it and, based on the terminal occurrences, let the structural analysis decide, the structure of which formula fits the input image best and how does the image segmentation looks like.

Terminals are detected using an OCR tool and a suitable strategy that chooses rectangular areas of the image for evaluation. For example, up to five terminals can be detected in the first formula in Figure 2: variables T and I, number 2 and a fraction line that can be interpreted as a minus sign as well. Each of the occurrences is assigned by a penalty (computed by the OCR tool) determining quality of the recognized symbol. The terminals, represented by their bounding boxes, labels and penalties, are processed by a grammar-based structural analysis. During a bottom-up parsing process, bigger rectangular areas labeled by grammar non-terminals are being derived, each derived area being assigned by a penalty again. Details on the recognition will be given in the following sections.

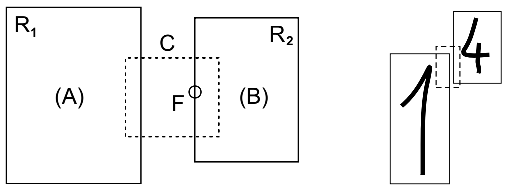

Figure 4. General production scheme for $N \rightarrow A \oplus B$ and production usage example.

3.1 Extension of 2D CF Grammars

We use an extension of 2D CF grammars to model relationships between mathematical symbols. By a *region* we mean a rectangular area in the input image. Let $N \rightarrow A \oplus B$ denote a production of our 2D CF grammar extension. The interpretation of the production is as follows. Regions labeled A and B can be united, producing a region labeled N. We do not require the regions to touch each other, they can even overlap. Permitted mutual positions of the regions are defined by a (spatial) *constraint* that is connected to the production. The form of the constraint is depicted in Figure 4.

R_1 and R_2 are two regions in the image, F is the center of the left border side of R_2. Constraint $\mathcal{C}$ assigned to the production always consists of a rectangular area the size and the position of which are given relative to R_1 and some significant point in R_2 (called *feature point*). In our example, the rectangle is C, while the feature point is F. The considered production can be applied when the following conditions are fulfilled:

- R_1, resp. R_2 can be labeled by A, resp. B.
- F is located inside C.

The resulting region is the smallest rectangle containing R_1 and R_2. As for the feature points, we usually use corners and centers of border sides. We also compute the baseline (when a new region like a fraction line is derived) and take the intersection with the left or the right border. Figure 4 shows usage of a production to model 'power to' relationship where the constraint is defined for the bottom-left corner of the region storing the symbol four.

Let $c(p)$ denote a gray-scale value (color) of pixel p. We consider $c(p)$ to be an integer in $\{0, \ldots, n\}$, where 0, resp. n denotes black, resp. white color. Let T be a set of pixels. We define *pixels penalty* of T as

$$\text{pix}(T) = \sum_{p \in T} (n - c(p))$$

The penalty of the derived region R_3 is computed as a sum of the following components:

- Penalty for the used 2D grammar production.
- Penalties assigned to R_1 and R_2.
- $\text{pix}\,(R_3 \setminus (R_1 \cup R_2))$
- Penalty for relative sizes and positions of R_1 and R_2.

To finish the description of the used 2D grammar, we will make two remarks. First, we have defined the constraint on the location of R_2, assuming we know the location and size of R_1. In our structural analysis, we will also need the opposite case, i.e., seeking R_2 while knowing R_1. We extend the production form by attaching one more constraint C' of the mentioned meaning. The second remark, in some cases of formulae constructs, it is more convenient to compose a new region from three regions instead of two (consider, e.g., a variable with a subscript and a superscript). This can be easily supported by introducing productions of the form $N \rightarrow A_1 \oplus A_2 \oplus A_3$.

3.2 Implementation

Our software consists of two independent layers. The first layer performs the *terminals detection*, while the second one is responsible for the *structural analysis*. The used structural analysis is driven by a 2D grammar defining supported mathematical formulae. The parsing algorithm is of a general nature and it can be used to recognize another types of structures but mathematical formulae if supported by a proper grammar.

3.2.1 Terminals Detection

We have developed an OCR tool for classification of image regions. The tool is based on a simple extraction of *features* from the input picture, which allows to recognize symbols varying in size. The k-nearest neighbor *classifier* is implemented to classify the extracted vectors. The tool is trained for two sets – handwritten and printed symbols.

A combination of *two strategies* for terminal symbols detection is employed. The *first strategy* works with *rectangular scanning windows* of some specific sizes. For example, we have a window to detect subscripts and another window to detect variables and numbers. Each of the windows is being moved trough the input image. Whenever it is in a new position, its content is evaluated by the OCR tool (provided that the pixels penalty of the view exceeds some defined threshold). The result of the evaluation is a set of terminals assigned by a penalty, where the penalty corresponds to the belief that the scanning window stores a particular symbol. Only the terminals with a sufficiently low penalty are included in the result.

Sizes of the windows are determined by expected sizes of symbols in the formulae which means the strategy requires tuning for typical inputs. It would be ideal in theory if we could use views of all sizes to scan the image, however, this approach is time expensive. Limiting the sizes of used views results in an acceptable performance but can miss some symbols in the image when their size differs from the expectation. Because of this we have added the *second strategy* based on preprocessing the content of the image by computing *connected components*. Selection of additional views for the evaluation is driven by these components. The bounding rectangles of the following areas are chosen: the components themselves, divisions of the components – up to two splitting horizontal and vertical points are considered, combinations of neighboring components.

3.2.2 Structural Analysis

We describe the algorithm that is used for the structural analysis phase. To simplify the description, we do not explain how information needed to track feature points and derivation trees is being updated. Just note that whenever a region R is labeled by N, we record by which production this was derived. This is exactly information needed to build the derivation tree.

1. Let $\mathcal{R}$ be a list of triples (R, l, p), where R is a region, l label assigned to R and p penalty of this assignment. Initialize $\mathcal{R}$ by results of the terminals detection phase for the input image I.

2. Iterate through $\mathcal{R}$. Let $(R, l, p) \in \mathcal{R}$ be the current element. For each production $N \rightarrow A \oplus B$ such that $l = A$ or $l = B$, take the rectangular area C defined by the proper production constraint and find subset $\mathcal{S} \subseteq \mathcal{R}$, where for each $(R', l', p') \in \mathcal{S}$ the production can be applied on R and R'. Let $\overline{R}$ be the derived region, $\overline{p}$ penalty of the derivation. If $\overline{p}$ is greater than some threshold then continue by the next iteration, otherwise check whether $\overline{R}$ has been already labeled by N. If not then append $(\overline{R}, N, \overline{p})$ at the end of $\mathcal{R}$. If there is already $(\overline{R}, N, p_2)$ in $\mathcal{R}$ and $\overline{p} < p_2$ then remove $(\overline{R}, N, p_2)$ from $\mathcal{R}$ and append $(\overline{R}, N, \overline{p})$, otherwise ignore the new derivation.

3. For each region $R \in \mathcal{R}$ labeled by the initial non-terminal, increase the penalty assigned to R by $\mathrm{pix}\,(I \setminus R)$. Among these regions, find the resulting one with the lowest penalty.

To speed-up the algorithm, we use a data structure storing points in a plane, allowing it to effectively evaluate queries of the type: for a rectangle, return the points that are located inside the rectangle (so called *orthogonal range searching*). It is used in step 2 to search for all regions fulfilling the production's constraint. A suitable data structure allowing to search in time $\mathcal{O}(\log n)$ can be constructed [1].

Our conclusions on time complexity are based on experimental results, we do not give an exact formula since it depends on many factors, including the number of the terminals detected during the first phase. Compared to the generalized Cocke-Younger-Kasami algorithm for 2D CF grammars [10], the expected time complexity is lower because the algorithm does not process all rectangles in the input image.

4 Results

We have implemented the pilot study in Java. The implementation includes a user interface allowing to browse formulae images, run the recognition on them and display results. Except the results, the interface also provides information helping to understand and tune the process of structural analysis. For example, it is possible to query for all regions labeled by a specific nonterminal, for penalties of related derivations, etc.

After tuning productions' parameters and training the OCR tool, the implementation has been tested on two sets of formula images – handwritten and printed. Approximately one third of the handwritten formulae covered the problematic cases we have discussed in section 2. The following table summarizes experimental results.

	total images	total errors	OCR errors	correct rate 1	correct rate 2
handwr.	330	48	37	85.5%	96.2%
printed	210	14	10	93.3%	98.0%

We give two correctness rates. The correctness rate 1 is computed as the ratio between the total number of errors and the number of processed images. The correctness rate 2 is computed after excluding the images unrecognized due to the OCR tool failures. The reason is that OCR errors occurred relatively often in the case of handwritten formulae and the structural construction method should be evaluated without their impact (in fact, it is possible to replace our OCR tool by a better one). More details on the errors types and their frequencies are given in the next table.

	OCR	terminals detection	structural analysis	total
handwr.	37	6	5	48
printed	10	3	1	14

Errors done during structural analysis can be usually avoided by tuning grammar productions' parameters, possibly by relaxing constraints, but this can lead to an increase in time complexity.

We have performed additional experiments on formulae images corrupted by the additive Gaussian noise and impulsive noise. The OCR tool again influences the quality of results, but when the correct terminals are passed to the structural analysis, it works quite well.

As for the time complexity of the method, most of the image recognitions were responsive. However, we have also faced some cases where it lasted for 20 seconds (formulae containing many nested fractions). In such a case, it is again possible to trade off between the speed and the quality by changing constraints.

As the last remark on the results, we will point out a limitation we have faced. Not all symbols in a formula can be separated by rectangles. This is mostly case of handwritten formulae and it causes problems for the scanning windows strategy (note that square root sign does not fall in this category since, we have handled it by decomposing to more components that are separable).

5 Conclusion and Future Work

We have showed that the method of structural construction can be applied to the off-line mathematical formulae recognition. It is more suitable for printed formulae since they exhibit more regularities and the impact related to the rectangular regions limitation is not so high. On the other hand, it helps to solve many segmentation problems that occur in handwritten formulae.

Our main contribution to the area of formulae recognition is summarized in the following achievements:

- Real segmentation of the image is driven by the structural analysis (no error corrections are needed). We took the advantage of the rich formula structure.

- The structural analysis is robust. It is penalty oriented and searches for the formula structure that best matches the input image. It can deal with noise, including misplaced symbols.

- An extension of 2D CF grammars is powerful enough to express the formulae structure. It can be also effectively parsed (thanks to constraints defined via rectangles and the usage of data structures for orthogonal range searching).

Our ongoing work attempts to include statistical learning methods that can be applied to obtain etalons of terminal symbols and productions parameters. The learned etalons can improve the terminals detection phase, while the learned productions parameters will improve tuning of the grammar for a concrete typesetting or handwriting style of formulae (so far we have tuned these parameters manually).

Our plan is to adopt ideas of structural construction to implement on-line formulae recognition. Again, we would like to design a method where segmentation of strokes is driven by the structural analysis. It is of an advantage that the analysis works regardless the input type, thus it is sufficient to focus on strategies that select candidates for segmented symbols. The results achieved in the off-line recognition promise the proposed method will work in the on-line case as well.

6 Acknowledgement

We are greatly thankful to M.I. Schlesinger from the Ukrainian Academy of Sciences in Kiev for discussions on the issue. The research was supported by the EC projects INTAS 04-77-7347 PRINCESS, IIST-004176 COSPAL and by the Czech Ministry of Education under project 1M0567.

References

[1] M. Berg, O. Schwarzkopf, M. Kreveld, and M. Overmars. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, 2000.

[2] K.-F. Chan and D.-Y. Yeung. Mathematical expression recognition: a survey. In *IJDAR*, volume 3, pages 3–15, 2000.

[3] P. Garcia and B. Couasnon. Using a generic document recognition method for mathematical formulae recognition. In *Proceedings of the SPIE 1998*, number 2390, pages 236–244, Berlin, Germany, 1998. Springer-Verlag.

[4] V. Kiyko. Recognition of objects in images of paper based line drawings. In *Third International Conference on Document Analysis and Recognition*, pages 970–973, Montreal, 1995.

[5] S. Lavirotte and L. Pottier. Mathematical formula recognition using graph grammar. In *Proceedings of the SPIE 1998*, volume 3305, pages 44–52, San Jose, CA, 1998.

[6] E. Miller and P. Viola. Ambiguity and constraint in mathematical expression recognition. In *AAAI/IAAI*, pages 784–791, 1998.

[7] D. Průša. *Two-dimensional Languages*. PhD thesis, Faculty of Mathematics and Physics, Charles University, Prague, 2004.

[8] D. Průša and V. Hlaváč. 2d context-free grammars: Mathematical formulae recognition. In *Proceedings of the PSC '06*, pages 77–89, Prague, Czech Republic, 2006. ČVUT.

[9] B. Savchynsky, M. Schlesinger, and M. Anochina. Parsing and recognition of printed notes. In *Proceedings of the conference Control Systems and Computers*, pages 30–38, Kiev, Ukraine, 2003. in Russian, preprint in English available.

[10] M. Schlesinger and V. Hlaváč. *Ten lectures on statistical and structural pattern recognition*, volume 24 of *Computational Imaging and Vision*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2002.

[11] K. Toyozumi, N. Yamada, K. Mase, T. Kitasaka, K. Mori, Y. Suenaga, and T. Takahashi. A study of symbol segmentation method for handwritten mathematical formula recognition using mathematical structure information. In *17th International Conference on Pattern Recognition (ICPR'04)*, volume 2, pages 630–633, 2004.