MST-Based Visual Parsing of Online Handwritten Mathematical Expressions

Lei Hu
Department of Computer Science
Rochester Institute of Technology
Rochester, USA
lei.hu@rit.edu

Richard Zanibbi
Department of Computer Science
Rochester Institute of Technology
Rochester, USA
rlaz@cs.rit.edu

***Abstract*—We develop a Maximum Spanning Tree (MST) based parser using Edmonds' algorithm, which extracts an MST from a directed Line-of-Sight graph in two passes. First, symbols are segmented by grouping input strokes, and then symbols and symbol pair spatial relationships are labeled. The time complexity of our MST-based parsing is lower than the time complexity of CYK parsing with 2-D Context-Free grammars. Also, our MST-based parser obtains higher formula structure and expression rates than published techniques using CYK parsing when starting from valid symbols. This parsing technique could be extended to include n-grams or other language constraints, and might be used for other notations.**

***Keywords*-MST-based parsing, handwritten math recognition, Parzen windows, shape contexts**

I. Introduction

Math expressions are an essential part of scientific documents. Handwritten math expression recognition can benefit human-computer interaction, especially for educational applications, and can be used to support math-aware search engines or recognizing the structure and content of handwritten documents [1]. Math expression recognition consists of three main parts: symbol segmentation, symbol recognition and structural analysis. Many current math expression recognition systems use 2-dimensional Stochastic Context-Free Grammars (SCFG) to define an expression language, which is then used to parse handwritten strokes or candidate symbols using the Cocke-Younger-Kasami (CYK) algorithm. Production rules in context-free grammars for handwritten math need to be designed carefully for specific domains, and often require special cases to deal with different writing orders. Sometimes the number of rules can be in the hundreds. Currently in practice, grammar creation is normally done manually based on an expression sample.

In this paper, we consider whether a more intuitive parsing technique based on visual features can compete with these grammar-driven parsers. Motivated by the observation that mathematical expression recognition can be posed as searching for a Symbol Layout Tree (SLT) representing symbols and their associated spatial relationships in a graph of handwritten strokes, we propose MST-based parsing with Edmonds' algorithm [2], and introduce new visual features based on shape contexts for use in parsing.

II. Related Work

For online handwritten math recognition, syntactic pattern recognition methods, and particularly CYK parsing with 2-dimensional Stochastic Context-Free Grammars (SCFG) are widely used, and obtain good performance. Context-free grammars can incorporate people's prior knowledge and heuristics well. With cost or probability estimates (e.g., for symbol classification confidences), search algorithms can be constructed to automatically define one or more best interpretations within the very large space of possible symbol segmentations, classifications, and structural relationships. The quality of grammars is key for these systems, as they constrain the output space: fewer possible interpretations lead to a higher probability of selecting the correct interpretation. However, 'ungrammatical' interpretations cannot be returned without additional processing. The time complexity of the CYK algorithm is $O(N^3|G|)$, where N is the number of symbols in the expression, and $|G|$ is the number of rules in the grammar.

Chou [3] presented the first Cocke-Younger-Kasami (CYK) algorithm adapted for 2-dimensional SCFGs, used for parsing typeset math images. Recognition has two steps: first a list of candidates characters in the image is created using template matching, and character candidates are then passed on for parsing. An 2-D extension of the CYK algorithm is used to perform maximum likelihood parsing in a manner similar to the Viterbi algorithm for Hidden Markov Models. The Inside/Outside is used to learn the production rule probabilities from a training set. In this paper, the system takes an entire binary image as a sentence, and then parses it hierarchically all the way down to the pixel level.

Yamamoto et. al [4] present an online handwritten math expression recognition system based on a SCFG at the stroke level. They represent the generation of the 2-D layout and appearance for handwritten strokes in formulae. CYK is used to identify the expression(s) derivable from the expression grammar that have the highest likelihood strokes and relationships. Each stroke is initially classified as a symbol, and this defines the stroke likelihood. The structure likelihood is calculated using geometric features.

Similar to [4], Le et. al [5] represent mathematical ex-

pressions using CFGs at the stroke level, and employ CYK for parsing. As for Yamamoto et al., these grammars were designed manually. Parsing tries to locate the legal expression with the highest combined probability for segmentation scores, symbol classification scores, structure scores, and production rule scores. SVMs are trained separately for two spatial relation groups: (Above, Below, Inside) and (Horizontal, Superscript, Subscript). The separation probability for each pair of adjacent strokes is computed, and used to define all possible symbol segmentation hypotheses. As a result, they need to create production rules to capture all possible writing orders, such as those before and after fractions, roots, and parentheses to avoid parsing failures. In symbol classification, they define specific rules for dot/comma.

Celik et. al [6] present a math expression recognition system based on 2-D context-free probabilistic graph grammar at the symbol level. The probabilistic graph grammar allows the system to find all valid interpretations and their associated probabilities according to the grammar. The likelihood of an interpretation depends upon the suitability of the symbols' spatial distribution for the production rules used, and likelihoods for recognized symbols. A drawback of graph grammars is that they are much more expensive to parse due to the complexity of subgraph matching, but they also provide production rules that are easier to read than CFG-based expression grammars.

Alvaro et. al [7], [8] present an online handwritten math expression recognition system using SCFGs defined at the symbol level. The paper [7] defines a 2D extension of SCFG and the corresponding version of the Cocke-Younger-Kasami (CYK) parsing algorithm. The SCFG determines whether to merge two symbol candidates or not based on the class label and class confidence produced by the symbol classifier. The probability associated with the grammar rule is based on statistics observed in the training data. Parsing contains two steps. First, initialization begins by building basic units (lexical units) from the set of symbol segmentation hypotheses. Next, the parser calculates new subproblems of increasing size, where both spatial and syntactic constraints are taken into account for each new subproblem. During parsing, there is a level for each subproblem size, and these levels store a set of elements which contain their two-dimensional structure information.

Awal et. al [9] present a math expression recognition system using multiple two dimensional context-free grammars. The two dimensional math expression grammars are combined by two sets of one dimensional grammars that consider the horizontal and vertical directions separately. Each production rule is associated with a spatial relation describing the layout between the elements of the rule, and each relation has an associated cost function. Two features, position and size difference are used to define costs. Gaussian models are constructed for each element of a relation. Given the input expression, a hypothesis generator produces all possible symbol candidates which will be passed to a symbol classifier and structure analyzer. The maximum number of hypotheses, strokes per symbol, and distances between strokes forming a symbol are restricted. For each symbol candidate, a score is assigned based on the symbol recognition cost and structural cost. The output of the recognition system is a relation tree similar to an operator tree. Each relation tree has its cost which is calculated recursively based on symbol classification and relationship probabilities. The system chooses the expression interpretation with minimum cost according to the expression grammars.

Simistira et. al [10] propose a CYK-based algorithm to parse handwritten math expressions. They use seven geometrical features and train a probabilistic SVM classifier to recognized spatial relations between two symbols or subexpressions. This paper assumes the symbols have been correctly recognized and symbols are divided into three categories: ascenders, centered and descenders. In addition, the chronological order of symbols is used during parsing.

MacLean et. al [11] use CFGs to generate all identifiable parses for the input strokes. Parses are represented as a fuzzy set, in which the membership grade of a parse measures the similarity between the expression and the handwritten input. Parsing contains two steps: shared parse forest construction and parse tree extraction. To identify and report parses efficiently, they adapt and apply rectangular partitions and shared parse forests, and introduce relational classes and interchangeability. A Bayesian scoring model was introduced in later work [12].

MST-Based Recognition. Graph-based parsing does not require the definition of a grammar, instead seeking to identify a subgraph with minimal cost or maximum probability subject to certain constraints. For math recognition, these have been based on Minimum Spanning Trees (MSTs).

Suzuki et. al [13] present a printed math expression recognition system using virtual link networks. Each node represents a symbol, and each edge is represented by four components: (parent candidate, child candidate, label, cost). Recognition is performed by finding a spanning tree for the network with minimum penalty. The penalty of the spanning tree contains two parts: local penalty and global penalty. The local penalty is calculated based on the distribution of relative sizes and positions for each relation type in parent-child links. The global penalty is based on predefined rules designed to exploit context. The total penalty is the sum of edge link penalties and the global penalty.

Matsakis [14] defined a segmentation method based on minimum spanning trees (MST). For the given expression, it will form an MST over the strokes. In the MST, each node represents a stroke, and the distance between any two strokes is the Euclidean distance between the centers of the bounding boxes. The segmentation method only considers partitions that form connected subtrees in the MST.

In the next section we introduce a new, more robust MST-

based parsing technique, which does not require a grammar.

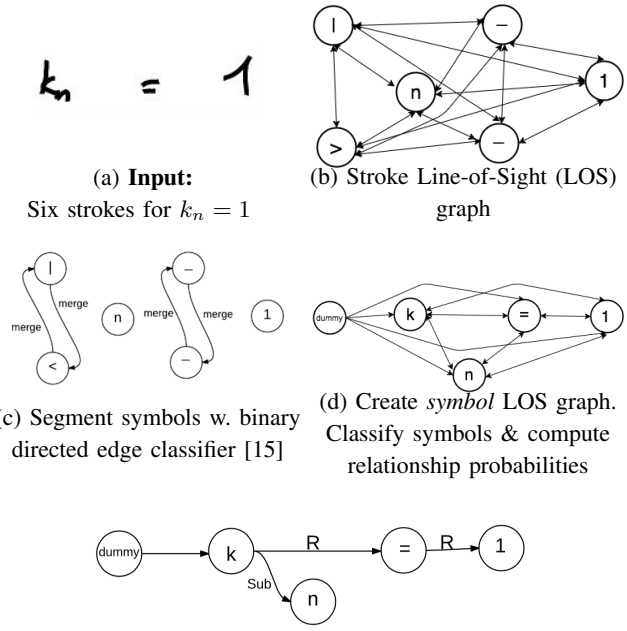
III. METHODOLOGY

MST-Based Parsing. Figure 1 shows an example of an MST-based parse. A Line-of-Sight (LOS) graph is constructed based on whether an unobstructed line can be drawn from the center of one stroke to the convex hull of another stroke [15]. After constructing the stroke level LOS graph [16] (see Figure 1 (b)), a binary classifier labels each edge as 'merge' or 'split' (Figure 1 (c)). Each connected component defined by 'merge' edges is taken to be a symbol candidate. A *symbol* level Line-of-Sight graph is then constructed using the symbol candidates, using the same LOS algorithm but with symbols rather than strokes as input (Figure 1 (d)). For each edge (symbol pair) in the symbol level Line-of-Sight graph, a symbol spatial relationship classifier produces a score list (a ranked list of scores (C_i, S_i) for each class C_i with score S_i). Edmonds' algorithm first adds a dummy node to the symbol LOS graph with all symbols as its children, which will allow us to represent the leftmost symbol on the main baseline of the expression. The algorithm then extracts a spanning tree at the symbol level (Figure 1 (e)). We then remove the dummy node, producing a Symbol Layout Tree.

Our method is quite different from Eto and Suzuki's MST-based algorithm [13]: they recognized math images, used undirected MSTs along with local beam searches, defined heuristic penalty functions, and the construction of their initial graph over symbols is unclear.

Edmonds' Algorithm. We apply Edmonds' algorithm [2] to a directed Line-of-Sight graph over given or recognized symbols [15], with symbol relationship classification probabilities associated with edges. Given this directed graph with real-valued edge weights, Edmonds' finds a rooted spanning tree with *directed* edges as output (see Figure 1 (d-e)). This fits our needs well, as the Symbol Layout Trees we wish to recover are also rooted directed trees - while more familiar, Prim and Kruskal's MST algorithms produce undirected spanning trees. We make one small change, in that we use Edmonds' to obtain the *maximum* probability spanning tree. Edmonds' algorithm greedily selects the incoming edge with the highest weight for each node, allowing the 'best' relationships to be selected from any location in the expression as the algorithm progresses. As the MST is constructed, if no cycles exist, then all selected edges form an MST. If there is a cycle, the algorithm contracts the cycle into a single node, and then recalculates edge weights going into and out the cycle before selecting the next edge.

We implement Edmonds' Algorithm as described in [2], with $O(|E| \times |V|)$ time complexity. In the worst case, $|E| = |V| \times (|V| - 1)$. Therefore, the time complexity of MST-based parsing is $O(|E| \times |V|) = O(|V|^2 \times |V|) = O(|V|^3) = O(N^3)$, where N is the number of symbols in the expression. We use symbol level Line-of-Sight graphs as input, with $|E|$ seldom reaching this worst case. On average,



(a) **Input:** Six strokes for $k_n = 1$

(b) Stroke Line-of-Sight (LOS) graph

(c) Segment symbols w. binary directed edge classifier [15]

(d) Create *symbol* LOS graph. Classify symbols & compute relationship probabilities

(e) Apply Edmonds' algorithm to obtain MST. **Output:** Remove dummy node for Symbol Layout Tree

Figure 1: MST-based Parse for a Handwritten Formula. In (d) visual and geometric features are used to compute symbol and relationship class probabilities independently

the number of edges in LOS is 3.3 times the number of strokes [16]. One could also use Gabow et al.'s [17] faster version with running time $O(|E| + |V| \log |V|) \in O(N^2)$.

Stroke Pair and Symbol Pair Relation Classification. We use random forests for both the binary stroke pair classifier ('merge'/'split'), and symbol pair relationship classifier (seven classes). The set of spatial relationship is one seven classes: *undefined (no-relation, _), Right (R), Subscript (Sub), Superscript (Sup), Above (A), Below (B), Inside (square root, I)*. Both classifiers also use Parzen window Shape Context features (described below), along with geometric features. In the binary stroke classifier we also use a time gap feature: for stroke pair (i, j), time gap feature is $j - i$.

Geometric features include distances between bounding box centers, distances between averaged centers (centers-of-mass), maximal point pair distances (for two points, one from each stroke), horizontal offset between the last point of the current stroke and first point of the next stroke, vertical distance between bounding box centers, writing slope (angle between the horizontal and the line connecting the last point of the current stroke and the first point of the next stroke) and writing curvature (angle between the lines between the first and last points of the current and subsequent stroke). We normalize all the geometric features in $[0, 1]$ except parallellity, writing slope and writing curvature.

Parzen window Shape Context Feature (PSC). Belongie et al. [18] propose shape contexts for shape matching and object recognition. The shape context [18] at a given point captures the distribution of the other points relative to it, and therefore provides a globally discriminative characterization. Shape context features have been usefully applied to math symbol classification [19]–[21], math symbol retrieval [22] and spatial relationship classification [23]. These shape context features are histograms, obtained by counting the number of points within each bin of a polar histogram. Therefore, all these previously used features are discrete, with each point only affecting the bin to which it belongs.

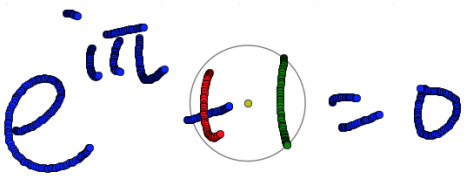
In our work, we compute shape contexts from foreground pixels in binary images generated from the stroke data [15], using a fixed image height of 200 pixels, and then adjusting the width to match the aspect ratio of the original expression. We have designed a continuous shape context feature based on Parzen Windows that insures every point contributes to all bins in the histogram, making the resulting distributions smoother. Parzen-window density estimation is a data-interpolation technique [24]. A kernel function is placed at the location of each point, generating 'pulses' that collectively define a continuous distribution. To produce our PSCs, these pulses are measured at the center point of each shape context bin and added to produce a histogram of real values that is then normalized. We use a two dimensional gaussian distribution as the Kernel function:

$$P(x,y) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} exp\left(-\frac{(x_i - x)^2}{2\sigma^2}\right) \frac{1}{\sqrt{2\pi}\sigma} exp\left(-\frac{(y_i - y)^2}{2\sigma^2}\right)$$

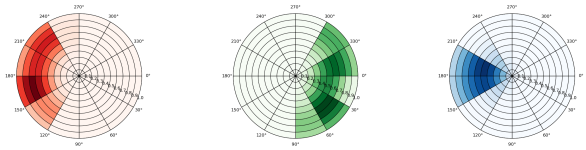
(x_i, y_i) are the coordinates of the points, while (x, y) represents the center of a given bin. As can be seen in Figure 2, the result is a simplified, radially skewed version of the input image.

Given a pair of strokes or symbols (i, j), the center of shape context is the average of their two bounding box centers. The radius can vary - we crop the histogram around pairs of symbols, but allow PSCs for strokes to be larger in order to include additional context, which we found empirically to be beneficial. The number of bins is the product of the number of angles (M) and distances (N). The shape context is divided into bins uniformly by angle and distance, as done by Álvaro [23]. We actually use three separate PSCs to characterize the distribution of points from each of our three sources (parent, child, and other). This produces $3MN$ features.

Figure 2 shows an example of stroke-level PSC. Different color represents different sources: red for points from the parent stroke, green for points from the child stroke, and blue for points from other strokes. The parent stroke (red) is the vertical stroke of '+' and child stroke (green) is '1', and other strokes are blue. Brightness indicates densities in the bins.



Expression with PSC center and perimeter shown



Parent stroke Child stroke Other strokes

Figure 2: Example of Parzen Shape Contexts. Here a directed edge between the vertical stroke of the '+' and stroke for the '1' are to be classified, making the '+' stroke the parent, and the '1' the child of the relationship. The center of the Shape Context is the average of the two stroke bounding box centers. Each PSC has 120 bins, with the PSC radius reaching the furthest parent or child stroke point (pixel). In experiments we use only 30 bins (5 distances $\times$ 6 angles). For symbol-level PSCs, we use the same representation after grouping all points from parent symbol strokes together, and all points from child symbol strokes together.

IV. EXPERIMENTAL RESULTS

Datasets. The Competition on Recognition of On-line Handwritten Mathematical Expressions (CROHME) is a well-established annual competition [25]. The results reported in this section are computed using the CROHME 2012 and 2014 data sets. CROHME 2012 has 1336 training and 486 test expressions; CROHME 2014 contains more structurally complex formulae, and is larger with 8834 training and 986 test expressions.

Training. Greedy search and cross validation are used to determine the parameters of Parzen window shape context feature [16]. We use 30 bins, with 5 angles and 5 distances. We use a Parzen window width of $\frac{1}{5}$ of the polar histogram radius. For symbol pairs, the shape radius is the longest distance from points in the two symbols to the polar histogram center, while the shape radius is 1.5 times of the longest distance for stroke pairs.

Symbol Segmentation and Classification. Classification rates for identifying LOS stroke graph edges as 'merge' or 'split' are quite high. Using the CROHME Test sets we obtain rates of 98.26% for 2012 and 97.88% for 2014. Symbol segmentation rates (Recall) are also strong, with 94.87% for 2012 and 92.41% for 2014. These Recall rates are within 1 percent of the best published results for systems trained using the CROHME Training set. The 2014 F-measure we obtain (92.43%) was the highest published value at the time of submission. It is interesting that these strong

Table I: Parsing Results with Provided Symbols (CROHME 2012 & 2014). Here, correct symbols are provided as input to parsing algorithms. Shown are rates for correct detection of relationships on edges (*Det.*), and correct detection and labeling of relationships (+*Class*). We also provide rates for correctly recognized formula structure (*Str.*), and formulae with correct structure and relationship labels (+*Class*)

	Rel. Rates		Formula Rates	
	Det.	**+Class**	**Str.**	**+Class**
2012 Test				
MST-based	96.16%	93.16%	**72.63%**	**67.70%**
Simistira et al. [10]			57.41%	56.37%
2014 Test				
MST-based	95.51%	91.08%	76.67%	67.44%

Table II: Parsing Results for Handwritten Strokes (CROHME 2014). Recognition rates shown require both correct segmentation or structure recognition, and correct classification. G: system uses a grammar. MST-based results are shown in bold

	Recognition Rates			
	Symbols	**Relationships**	**Expressions**	**G**
MyScript*	93.91%	94.26%	62.68%	✓
Alvaro [7]	86.59%	84.23%	37.22%	✓
MST-based	**81.95%**	**79.89%**	**26.88%**	
Awal et. al [9]	76.53%	71.77%	26.06%	✓
Le et. al [5]	69.72%	66.83%	25.66%	✓
Hu et. al	76.64%	70.78%	18.97%	
Yao and Wang	78.45%	61.38%	18.97%	
Aguilar [27]	66.97%	60.31%	15.01%	✓

** large, independent training set used. Others use CROHME 2014 Train*

results are obtained without the use of OCR or expression grammars. Additional details are available elsewhere [15]. For symbol classification we used Davila and Zanibbi's classifier [26], which obtains a classification rate of 88.6% for the CROHME 2014 Test data.

Symbol Relationship Classification. When classifying spatial relationships between symbols, there are seven classes for CROHME SLTs: *undefined (no-relation), Right, Subscript, Superscript, Above, Below and Inside (square root)*. For training and testing, we generated samples for all pairs of symbols in CROHME expressions. For the CROHME Test sets we obtain recognition rates of 97.64% for 2012, and 96.55% for 2014.

Parsing from Provided Symbols. Table I shows results when parsing with symbols provided from Ground Truth. For our algorithm this means starting from step (d) in Figure 1, but with correct symbol classes given.

For comparison, we provide results from a CYK parsing-based system using Stochastic Context-Free Grammars (SCFG) and a manually designed expression grammar [10]. Their symbol relationship classifier is SVM-based, with an error rate of 2.8% for CROHME 2012. This is close to our symbol pair relation classifier error rate, but unlike Simistira et al. symbol classes are not used to compute relationship classification features in our system. The MST-based parsing system obtains a 15% higher formula structure recognition rate, and 10% higher expression rate than Simistira et al., despite having a lower isolated symbol relationship recognition rate when correct symbols are provided as input.

When symbol labels are used, we can obtain nearly a 2% improvement in structure and expression rates simply by filtering invalid labels for symbol pairs with associated relationships before applying Edmonds' algorithm [16].

Parsing from Strokes. Table II compares the MST-based parser with other results obtained for the CROHME 2014 Test set. MyScript obtains the highest expression rate, but did so using a separate, much larger training set. For data sets trained using CROHME data, we obtain the third-highest symbol, relationships, and expression recognition rates after MyScript and Álvaro, but without the use of an expression grammar, and while classifying symbols and symbol relationships independently of one another.

Additional Results. Both symbol segmentation and structural analysis can be represented as edge labeling tasks in graphs over strokes [25]. We completed preliminary experiments with parsers and parser ensembles that segment symbols and parse relationships directly from stroke graph edges, and found that simultaneous segmentation and structural analysis achieved lower structure rates than doing structural analysis after symbol segmentation [16]. We also tested a backtracking version of the MST-based parser at the symbol level (i.e. after segmentation), considering the top-k symbol classes, and using a simple product of the three probabilities for each label combination (parent symbol, child symbol, relationship). The algorithm in Figure 1 outperforms this backtracking algorithm, possibly because it is a simple greedy algorithm.

V. Conclusion

We propose a new 'visual' MST-based algorithm for parsing handwritten mathematical notation. Edmonds' algorithm is used to obtain a directed MST representing symbol layout, with only visual and geometric features used to segment, classify, and determine relationships between symbols. Our visual features include new Parzen window modified Shape Contexts (PSCs), which have been shown to be effective for symbol segmentation and relationship classification. Edmonds' algorithm has lower time complexity ($O(N^2)$ for N strokes [17]) than widely-used CYK parsers with context-free grammars ($O(N^3|G|)$ for $|G|$ production rules). Experimental results demonstrate that our MST-based parsing obtains higher formula recognition rates than a published CYK-based parser when symbols are provided [10], and is very competitive parsing from handwritten strokes for CROHME 2014 data [25].

The fact that MST-based parsing does not require grammars is appealing. Syntactic pattern recognizers require

a new grammar to be defined when the target language changes. In this regard MST-based parsing has better generalization ability than Stochastic Context-Free Grammars for other notations. That said, our method may benefit greatly from n-grams or other language constraints. In the future, more research is needed on how to integrate symbol segmentation and classification into MST-based parsing. Some of the techniques presented in this paper might also be incorporated into SCFG-based parsers (e.g. PSC features).

ACKNOWLEDGMENT

This material is based upon work supported by the National Science foundation under Grant No. IIS-1016815.

REFERENCES

[1] R. Zanibbi and D. Blostein, "Recognition and retrieval of mathematical expressions," *International Journal on Document Analysis and Recognition*, vol. 15, no. 4, pp. 331–357, 2012.

[2] J. Edmonds, "Optimum branchings," *Journal of Research of the National Bureau of Standards B*, vol. 71, no. 4, pp. 233–240, 1967.

[3] P. A. Chou, "Recognition of equations using a two-dimensional stochastic context-free grammar," in *Intelligent Robotics Systems Conference*, 1989, pp. 852–865.

[4] R. Yamamoto, S. Sako, T. Nishimoto, and S. Sagayama, "On-line recognition of handwritten mathematical expressions based on stroke-based stochastic context-free grammar," in *International Workshop on Frontiers in Handwriting Recognition*, Oct. 2006, pp. 249–254.

[5] A. D. Le, T. V. Phan, and M. Nakagawa, "A system for recognizing online handwritten mathematical expressions and improvement of structure analysis," in *International Workshop on Document Analysis Systems*, Apr. 2014, pp. 51–55.

[6] M. Celik and B. Yanikoglu, "Probabilistic mathematical formula recognition using a 2d context-free graph grammar," in *International Conference on Document Analysis and Recognition*, Sep. 2011, pp. 161–166.

[7] F. Alvaro, J. Sanchez, and J. Benedi, "Recognition of on-line handwritten mathematical expressions using 2d stochastic context-free grammars and hidden markov models," *Pattern Recognition Letters*, vol. 35, pp. 58–67, 2014.

[8] F. Alvaro, J. Sánchez, and J. Benedí, "An integrated grammar-based approach for mathematical expression recognition," *Pattern Recognition*, vol. 51, pp. 135–147, 2016.

[9] A. Awal, H. Mouchère, and C. Viard-Gaudin, "A global learning approach for an online handwritten mathematical expression recognition system," *Pattern Recognition Letters*, vol. 35, pp. 68–77, 2014.

[10] F. Simistira, V. Katsouros, and G. Carayannis, "Recognition of online handwritten mathematical formulas using probabilistic svms and stochastic context free grammars," *Pattern Recognition Letters*, vol. 53, pp. 85–92, 2015.

[11] S. MacLean and G. Labahn, "A new approach for recognizing handwritten mathematics using relational grammars and fuzzy sets," *International Journal on Document Analysis and Recognition*, vol. 16, no. 2, pp. 139–163, 2013.

[12] S. MacLean and G. Labahn, "A bayesian model for recognizing handwritten mathematical expressions," *Pattern Recognition*, vol. 48, no. 8, pp. 2433–2445, 2015.

[13] Y. Eto and M. Suzuki, "Mathematical formula recognition using virtual link network," in *International Conference on Document Analysis and Recognition*, Sep. 2001, pp. 762–767.

[14] N. Matsakis, "Recognition of handwritten mathematical expressions," Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, May 1999.

[15] L. Hu and R. Zanibbi, "Line-of-sight stroke graphs and parzen shape context features for handwritten math formula representation and symbol segmentation," in *Proc. ICFHR*, 2016.

[16] L. Hu, "Features and algorithms for visual parsing of handwritten mathematical expressions," Ph.D. dissertation, Rochester Institute of Technology, 2016.

[17] H. N. Gabow, Z. Galil, T. Spencer, and R. E. Tarjan, "Efficient algorithms for finding minimum spanning trees in undirected and directed graphs," *Combinatorica*, vol. 6, no. 2, pp. 109–122, 1986.

[18] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 4, pp. 509–522, 2002.

[19] L. Ouyang, "A Symbol layout classification for mathematical formula using layout context," Master's thesis, Rochester Institute of Technology, Rochester, NY, 2009.

[20] L. Ouyang and R. Zanibbi, "Identifying layout classes for mathematical symbols using layout context," in *IEEE Western New York Image Processing Workshop*, 2009.

[21] N. S. T. Hirata and W. Y. Honda, "Automatic labeling of handwritten mathematical symbols via expression matching," in *International Conference on Graph-based Representations in Pattern Recognition*, May 2011, pp. 295–304.

[22] S. Marinai, B. Miotti, and G. Soda, "Using earth mover's distance in the bag-of-visual-words model for mathematical symbol retrieval," in *International Conference on Document Analysis and Recognition*, Sep. 2011, pp. 1309 –1313.

[23] F. Alvaro and R. Zanibbi, "A shape-based layout descriptor for classifying spatial relationships in handwritten math," in *ACM Symposium on Document Engineering*, Sep. 2013, pp. 123–126.

[24] N. Kwak and C.-H. Choi, "Input feature selection by mutual information based on parzen window," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 12, pp. 1667–1671, Dec. 2002.

[25] H. Mouchère, R. Zanibbi, U. Garain, and C. Viard-Gaudin, "Advancing the state of the art for handwritten math recognition: the crohme competitions, 2011–2014," *International Journal on Document Analysis and Recognition*, pp. 1–17, 2016.

[26] K. Davila, S. Ludi, and R. Zanibbi, "Using off-line features and synthetic data for on-line handwritten math symbol recognition," in *International Conference on Frontiers in Handwriting Recognition*, Sep. 2014, pp. 323–328.

[27] F. JulcaAguilar, N. Hirata, C. ViardGaudin, H. Mouchere, and S. Medjkoune, "Mathematical symbol hypothesis recognition with rejection option," in *International Conference on Frontiers in Handwriting Recognition*, Sep. 2014, pp. 500–505.