Modeling Flowchart Structure Recognition as a Max-Sum Problem

Martin Bresler, Daniel Průša, Václav Hlaváč
Czech Technical University in Prague, Faculty of Electrical Engineering
Department of Cybernetics, Center for Machine Perception
166 27, Praha 6, Technická 2, Czech Republic
{breslmar, prusapa1, hlavac}@cmp.felk.cvut.cz

***Abstract*—This work deals with the on-line recognition of hand-drawn graphical sketches with structure. We present a novel approach, in which the search for a suitable interpretation of the input is formulated as a combinatorial optimization task – the max-sum problem. The recognition pipeline consists of two main stages. First, groups of strokes possibly representing symbols of a sketch (symbol candidates) are segmented and relations between them are detected. Second, a combination of symbol candidates best fitting the input is chosen by solving the optimization problem. We focused on flowchart recognition. Training and testing of our method was done on a freely available benchmark database. We correctly segmented and recognized 82.7% of the symbols having 31.5% of the diagrams recognized without any error. It indicates that our approach has promising potential and can compete with the state-of-the-art methods.**

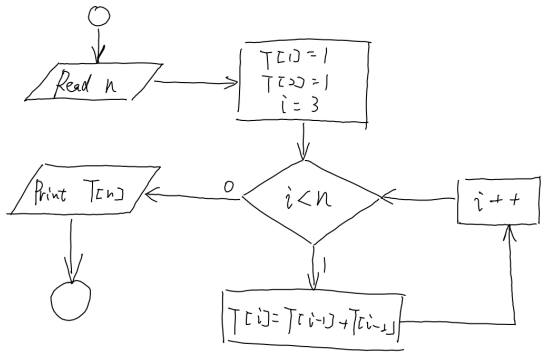


Fig. 1. Example of a flowchart from the FC database.

I. Introduction

This work deals with the on-line recognition of hand-drawn graphical sketches with structure. The topicality of such a task is implied by the proliferation of tablets and tablet PCs in recent years. To realize the recognition, two subproblems have to be solved: how to detect elementary units and how to expose hierarchical relationships over them. Here we benefit from the idea of structural construction paradigm presented by M.I. Schlesinger and V. Hlavac [1]. The segmentation phase does not make final decisions, it only identifies which groups of strokes can potentially form a symbol and in which possible local relationships the candidates can participate. It is up to the structural analysis to select the candidates that really fit into the whole pattern. One frequent approach to the structural analysis is to utilize a grammar. Several types of 2D grammars were proposed or adopted, e.g., to express the recursive character of mathematical formulas [2], [3], or the structure of diagrams [4]. Non-grammar based methods exploit a mixture of various statistical and discrete models and algorithms [5], [6], [7].

We introduce a method, in which the search for a suitable interpretation of the input is formulated as a combinatorial optimization task – the max-sum problem [8]. This give us a relatively large expressive power, at least when compared to other models like the minimum spanning tree finding presented in [5]. As a disadvantage, it may appear that a general max-sum problem is NP-hard. However, we show that solvers nowadays available are responsive enough on instances whose size safely covers the number of symbol candidates we generate for relatively large diagrams. It is even possible to use solvers for integer linear programming (ILP) since the max-sum problem can be easily transformed to it. We have verified that our method has a very good potential on diagrams and schemes. Flowchart diagrams were selected as the domain for our experiments. This choice was partly influenced by the FC database [9], which is freely available. There are documented methods applied to this database, together with corresponding results [9], [10], so we can compare the approaches. An example of a flowchart is shown in Figure 1. There are six symbol classes in the database: *arrow*, *connection*, *data*, *decision*, *process*, *terminator*. As for the other works focused on flowcharts, most of recently published methods put unnatural requirements on the user [11], [12]. For example, a pre-defined order of strokes or additional gestures are required. It is impossible to make a comparison with these methods. Also, we have been ignoring text so far. Nevertheless, it is straightforward to extend the model to work with text. Although, it would require to classify strokes into classes *text* and *shapes*. Recent text/non-text classifiers give promising results [13], [14].

The order of sections follows the recognition pipeline. Section II describes, how candidates for symbols are found. Section III explains, how hypotheses on relations among entities are created. Section IV shows, how the structure detection is formulated as a max-sum problem. Finally, the method is experimentally evaluated and its advantages are discussed in Section V.

II. Symbol Candidates Detection

The system of symbol candidates detection was introduced in our previous work [15]. Here we present briefly the basic

principles. The method is based on grouping neighboring strokes. Only those groups of strokes that are temporally and spatially compact are considered. Each admissible set of strokes is classified by a multiclass SVM classifier based on our own descriptor. The classifier provides three most probable candidates along with their confidence, a real number from $\langle 0, 1 \rangle$. Overall, the method finds 91.9% of symbols on the test dataset of the FC database while it generates 8.8 times more symbol candidates than is the number of symbols in a diagram in average.

A. Strokes Grouping

All possible sets of strokes, which fulfil the following conditions, are created: (1) the strokes are spatially close, (2) the set does not contain more than five strokes, (3) the set consists of two parts at most, where each of them was drawn consecutively. Two strokes are spatially close if the distance between their two closest points is smaller than the threshold $distThreshold = \alpha \cdot D_{\text{med}}$, where D_{med} is the median of values determined as lengths of diagonals over bounding boxes of all single strokes present in a diagram. The constant α has been empirically chosen to be 0.35. The requirements and α are derived from the data in the FC database, see Figure 2.

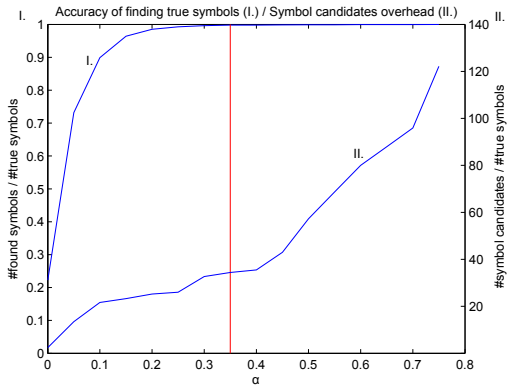


Fig. 2. The result of experiments over FC database to obtain an optimal value of the α coefficient. The graph shows the accuracy which improves as α grows (I.) and the growth of the number of symbol candidates (II.) at the same time. The choice of α is thus a tradeoff.

B. Classification

Each group of strokes generated in the previous step is classified by a multiclass SVM. The classifier uses our own descriptor of dimension 90. It is based on histograms of distances between points, angles between line segments given by neighboring points, and compositions (combinations of basic stroke elements). The classifier was learned on groups of strokes generated on annotated diagrams of the training dataset. Groups of strokes representing nothing (having no match in the annotated database) were used as negative examples and a new special class was created. Therefore, the classifier can reject a symbol candidate by classifying it as an instance of this class. We also fitted a logistic regression on the classifier response to obtain the posterior probability that a symbol candidate belongs to the recognized class.

III. Modeling Relations

To interpret the structure of a diagram, it is necessary to examine relations between entities. This extends the information which is utilized by the method to make decisions. Links between entities represented by arrows are the most characteristic feature of flowcharts. Our model of relations is based on connection points of particular entities.

A. Connection Points

We define four connection points (top, right, bottom, left) for each non-arrow entity and two connection points (head, tail) for each arrow entity. See Figure 3. These points identify where arrows can be connected to other entities. We propose three algorithms for computation of connection points where the usage depends on entity type.

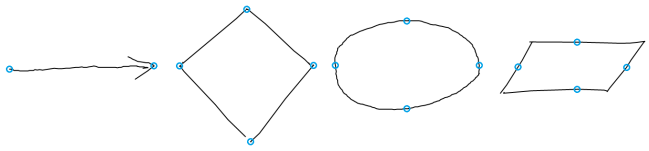


Fig. 3. Examples of connection points of an arrow, a decision, a terminator, and a data entity.

1) Arrow Entities: The strokes are split into the head and the shaft of the arrow. All possible splits, where the number of head strokes is not higher than four, are considered. The best split is chosen according to the scoring function, which is based on four requirements: (1) the bounding box of the head should not differ a lot from a square, (2) the distance between the center-point of the head bounding box and the closest end-point of the shaft should be small, (3) the curvature of the head strokes should be high, (4) the bounding boxes of the head and the shaft should not overlap. The direction of the head is determined according to the line segment given by the two shaft points closest to the head. The connection points of the arrow are determined for the best split. The point where the arrow is coming from (tail) is defined as the end-point of the shaft which has the greatest distance to the head. The point where the arrow is heading to (head) is defined as the one of the middle points of the bounding box edges, depending on the head direction.

2) Decision Entities: The left and the right connection points are defined as the points with the smallest distance on x-axis to the left and the right edge of the bounding box, respectively. The top and the bottom points are defined analogously, considering the distance on y-axis.

3) Other Entities: We cast four rays from the bounding box center to four directions (top, right, bottom, left). The connection points are defined as the intersections of the rays with either a stroke or an edge of the bounding box.

B. Relations Between Candidates

We define three types of relations between entity candidates: (1) conflict – two candidates share one or more strokes, (2) overlap – two non-arrow candidates has overlapping bounding boxes, (3) arrow connection - relations between an arrow and a non-arrow candidates. All possible pairs of

candidates are examined to find all relations. Potential conflicts are detected first. In the case there is no conflict, a relation of type (2) or (3) is detected depending on the types of both candidates. Let us note there is no relation defined between two arrows. This type of relation is not necessary, because overlapping arrows are suppressed later by the model.

Each relation gets a score which we denote s_1, s_2, and s_3 for relations of type (1), (2), and (3), respectively. Values of the score are defined as follows:

$$s_1 = -\infty, \tag{1}$$

$$s_2 = -S_{A\cap B}/\min(S_A, S_B), \tag{2}$$

where A and B are bounding boxes of the first and the second entity and S_A, S_B, and $S_{A\cap B}$ are surfaces of bounding boxes A, B, and their intersection, respectively.

$$s_3 = \exp\left(\frac{\ln(0.5)}{distThreshold} \cdot d\right), \tag{3}$$

where d is the distance between the two closest connection points of both entities and $distThreshold = \alpha \cdot D_{\text{med}}$ is a distance threshold when $s_3 = 0.5$. The threshold is the same as in Section II-A. Let us note that first two types of relations are negative (meaning we would like to prevent them from occurring), while only the third one is positive.

IV. Problem Formulation

So far, we have described how to obtain the symbol candidates and the relations between them, both assigned with a score expressing their confidence or support of the relations. The goal is to choose a combination of symbol candidates, giving the highest score when the involved particular scores are summed. This is an optimization task formulated here as an instance of the max-sum problem.

A. Max-Sum Formulation of the Basic Model

The pairwise max-sum labeling problem [8] (also known as the weighted constraints satisfaction problem) is defined as maximizing a sum of unary and binary functions (potentials) of discrete variables, i.e. as computing

$$\max_{\boldsymbol{k}\in K^V} \Big[\sum_{u\in V} g_u(k_u) + \sum_{\{u,v\}\in E} g_{uv}(k_u, k_v)\Big], \tag{4}$$

where an undirected graph $G = (V, E)$, a finite set K, and numbers $g_u(k_u), g_{uv}(k_u, k_v) \in \mathbb{R} \cup \{-\infty\}$ are given. We maximize over assignments of labels from K to nodes of G. Each node u and edge $\{u, v\}$ is then evaluated by the cost given by functions g_u and g_{uv}. In general, the max-sum problem has many applications, such as computing the MAP configuration of a Markov random field.

In our model, each symbol candidate and each positive relation defines a single node of the graph G. An edge is defined for each pair of interacting nodes (two symbol candidates in a negative relation, a symbol candidate and its positive relation, two positive relations of the same symbol candidate). Two labels are used, $K = \{0, 1\}$ where 0 means the candidate is not a part of the solution and 1 means it is. The numbers $g_u(k_u), g_{uv}(k_u, k_v)$ are set to express the score of symbol candidates and relations and to model natural restrictions as follows: $g_u(0) = 0$ and $g_u(1) = s$ for each symbol candidate or positive relation u with the confidence (score) s. Further, for all pairs of objects $(u, v) \in E$

1) $g_{uv}(1, 1) = -\infty$ if there is a conflict between objects u and v
2) $g_{uv}(0, 1) = -\infty$ if u is a symbol candidate and v is its positive relation (requiring the existence of u)
3) $g_{uv}(1, 1) = -\infty$ if u and v are both positive relations of the same arrow using the same connection point (arrow can come from or head to one point at most)
4) $g_{uv}(1, 1) = s$ if u and v are two non-arrow symbol candidates with overlapping bounding boxes where s is given by (2)
5) $g_{uv}(k, \ell) = 0$ in all other cases

A good commensurability of confidences and scores in the model is confirmed by our experiments. Note that we also tested a set up of the unary and binary potentials based on logarithms of confidences and scores, however, it did not lead to better results.

B. Adding Additional Constraints

A natural requirement for a completed flowchart is to have fully connected arrows (i.e., they do not come from or head to nowhere). We have found it good to include this into the model to increase the overall recognition accuracy. This is done by adding one auxiliary node for each arrow in the graph G. Such a node v represents the fact that the related arrow u is not fully connected. We set $g_v(0) = g_v(1) = 0$ and

1) $g_{uv}(1, 0) = g_{uv}(1, 1) = -2M$ where M is a suitable (large) constant, $0 \ll M \ll \infty$
2) $g_{vw}(0, 1) = M$ for each positive relation w of the arrow u

When an arrow node has the label 1, there is $-2M$ penalty no matter what is the label of its corresponding auxiliary node. The only way how to neutralize this penalty is to fully connect the arrow (i.e., there will be two positive relations of the arrow, each contributing M). Notice that there can not be more than two positive relations for one arrow, because each arrow has only two connection points, hence two positive relations using the same connection point of an arrow are necessarily in a conflict (resulting in $-\infty$ penalty). All the constructs applied in the model are illustrated by an example in Figure 4.

C. Solving the Optimization Task

The max-sum is a very general NP-hard optimization problem. Some special forms as submodular max-sum problems can be solved in a polynomial time [8]. Unfortunately, this is not our case. However, the size of graphs we generate is not so big (397 nodes and 3 223 edges in average). Therefore, general solvers are able to solve them fast (see Section V). We tested the max-sum solver Toulbar2 [16]. Its disadvantage is that it supports only non-negative integer costs. Therefore, we formulated the max-sum problem as the integer linear program and solved it using CPLEX library [17]. The conversion is done using the linear programming relaxation of the problem:

$$\max_{\boldsymbol{\mu}} \left[\sum_{k \in K} \sum_{u \in V} \mu_u(k) g_u(k) + \sum_{k,\ell \in K} \sum_{\{u,v\} \in E} \mu_{uv}(k,\ell) g_{uv}(k,\ell) \right] \quad (5)$$

$$\text{s.t.} \quad \sum_{k \in K} \mu_u(k) = 1, \qquad \forall u \in V,$$

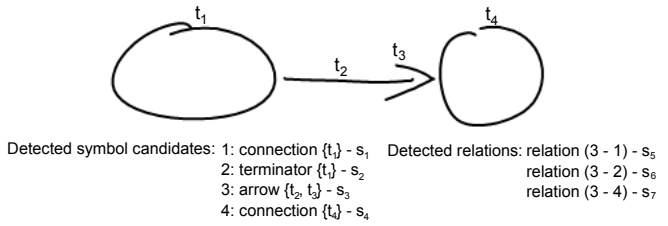
$$\sum_{\ell \in K} \mu_{uv}(k,\ell) = \mu_u(k), \qquad \forall \{u,v\} \in E \ \wedge \ \forall k \in K,$$

$$\mu_{uv}(k,\ell) = \mu_{vu}(k,\ell), \qquad \forall \{u,v\} \in E \ \wedge \ \forall k,\ell \in K,$$

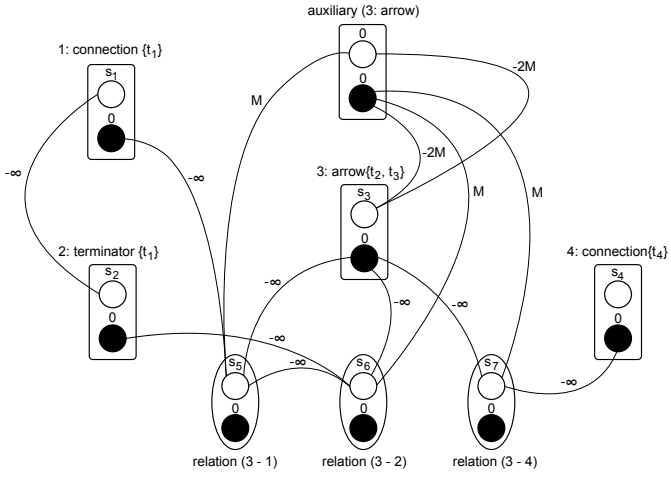
$$\boldsymbol{\mu} \geq \mathbf{0}.$$

If components of $\boldsymbol{\mu}$ are restricted to integral values only (i.e., they are 0 or 1), we obtain the desired ILP whose optimal value equals the optimal value of (4). The label $k_u \in K$ chosen in a node $u \in V$ is the only label satisfying $\mu_u(k) = 1$.

The ILP formulation give us a possibility to put more constraints (expressed as linear inequalities) to the model, e.g., each diagram must begin and end with a connector or a terminator symbol, i.e., all other non-arrow symbols must have at least one arrow coming in and coming out. However, we decided not to model such a restriction, because we want to be able to recognize unfinished sub-diagrams as well.



(a) A simple example of a flowchart consisting of four strokes and three symbols. The symbol detector found four symbol candidates, which are listed with their ids, sets of related strokes and confidences. Detected positive relations are listed with ids of the symbol candidates participating in the relation.



(b) The max-sum model corresponding to the example above. Rectangular nodes represent symbol candidates while ellipse nodes represent positive relations. Inside each node, both possibilities for labeling the node are represented (white circle – the label 1, black circle – the label 0). Each circle k in a node u is assigned by the value $g_u(k)$. An edge connecting a label k in a node u and a label ℓ in a node v is assigned by the value $g_{uv}(k,\ell)$. Edges with zero costs are not visualized.

Fig. 4. The flowchart (a) recognition modeled as a max-sum problem (b).

V. Experiments

We present results of experiments we performed on the test dataset, a part of the FC database. The set contains 127 samples consisting of 14.7 symbols and 37.2 strokes in average. We introduce a methodology used to match recognition results with the annotated diagrams. Correct rate and time performance of the recognition are given and the achieved accuracy is compared to the results reported by Lemaitre et al. [10].

A. Correct Rate and Methodology of its Measurement

The FC database provides annotations only for symbols, there is no information about relations. Users drew symbols by multiple strokes sometimes. Those redundant strokes are made to correct or beautify symbols. Therefore, it might happen that although a symbol is recognized correctly it does not comprise of exactly the same strokes as the annotated counterpart in the database (some redundant strokes are missing). We applied the following criterion to compare output of our recognizer with the annotated diagrams. We tried to match each annotated symbol with one of the recognized symbols. Two symbols match if they are of the same type and their bounding boxes overlap of 80%. The error rate is expressed as the number of unmatched annotated symbols.

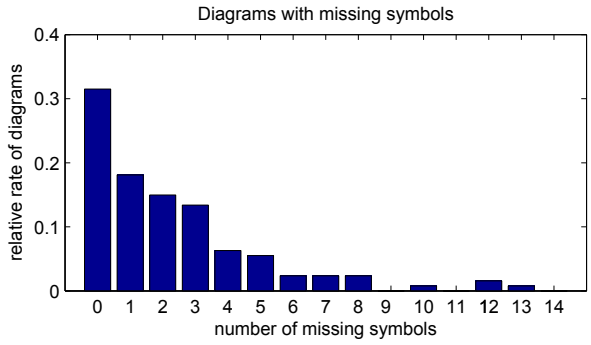


Fig. 5. The histogram lists the counts of diagrams by the number of missing symbols.

Overall, 31.5% of the diagrams were correctly recognized and about one half is recognized with one missing symbol at most. More details can be found in Figure 5. The accuracy of the first part of the pipeline was 88.7%. It means that there were 11.3% of the annotated symbols missing in the set of symbol candidates. After performing the optimization of the max-sum model, there were 17.3% of the annotated symbols missing. A bigger responsibility for errors in the recognition has the detection of symbol candidates. Therefore, we claim that the proposed max-sum model is a suitable tool for the diagram structure recognition.

Lemaitre et al. [10] developed a grammar based recognition method. They expressed its accuracy as the rate of correctly segmented and recognized symbols over the whole test dataset. Although the authors admit it is not the best way, they consider a symbol to be correctly recognized if the result consists of the very same strokes as the annotation. When we apply their stricter metric, our accuracy is 74.3%, which is still higher than 72.4% reported by the cited authors. We have to admit that their system deals with the text, which we are ignoring so far,

however, as we have already mentioned in the introduction, the text can be identified by embedding a text/non-text classifier.

B. Performance

We implemented the method in C# and tested it on a standard tablet PC Lenovo X61 (Intel Core 2 Duo 1.6 GHz, 2 GB RAM) and a desktop PC (Intel Core 2 Quad Q9550 2.83 GHz, 8 GB RAM) with 64-bit Windows 7 operating system both. CPLEX library [17] was used to solve the max-sum problem expressed as ILP in all experiments. Table I shows the results of time measurement (depending on the number of strokes).

	minimal	maximal	average	median
optimization	0.2 / 0.2	31.5 / 10.1	4.0 / 1.8	3.3 / 1.6
whole recognition	0.6 / 0.4	65.7 / 31.8	6.9 / 3.3	4.8 / 2.4

TABLE I. RUNNING TIME IN SECONDS.

The values confirm that the optimization is solved relatively fast. The time needed to recognize the whole diagram in the worst case can seem high. The impression improves if we express the average time per stroke, it is 0.167 / 0.080 seconds for the whole recognition and 0.101 / 0.046 seconds for the optimization. Moreover, the user spends even higher amount of time drawing the diagram, thus some time-consuming computations (especially features extraction) can be performed while the user is drawing. Since it was not our primary goal to make the recognition of symbol candidates as fast as possible, we think that after some tuning the time of this phase will be further reduced.

VI. CONCLUSIONS

We presented a novel method for the flowchart diagrams recognition consisting of two phases. Several mutually conflicting symbol candidates are detected by the first phase. Then, the best interpretation of the input is selected out of the candidates by solving the formed max-sum problem. We showed the method has a good potential. Despite that a large number of candidates is generated, the optimization performs well and causes less recognition errors than the detection of candidates and relations. This allows to concentrate on a better classification that will throw out a less number of ground truth symbols.

The experiments we performed on the freely available FC database confirm the statements given above. Our current implementation is not worse than the grammar based method presented by Lemaitre et al. The achieved accuracy already allows to develop an application recognizing (and beautifying) hand-draw flowcharts with a user's assistance. When the system makes an error the user selects the unrecognized symbol or relation and makes a correction. Since the frequency of errors is not high, the user will not be delayed by correcting too much. The successful usage of ILP solver gives additional possibilities for modeling structural relations. It is not necessary to be limited by the expressive power of the max-sum problem, we can go further and model additional features by linear inequalities of an integer linear program. This could help to represent more complex relationships exhibited for example by mathematical formulas. We think this fact would motivate other researchers to explore the method deeply on various domains.

ACKNOWLEDGMENT

The first author was supported by the Technology Agency of the Czech Republic under the project TE01020197. The second author was supported by the Grant Agency of the CTU under the project SGS13/205/OHK3/3T/13. The third author was supported by the EC project FP7-288553 CloPeMa.

REFERENCES

[1] M. I. Schlesinger and V. Hlaváč, *Ten Lectures on Statistical and Structural Pattern Recognition (Computational Imaging and Vision)*, 2002nd ed. Springer, Mar. 2012.

[2] E. G. Miller and P. A. Viola, "Ambiguity and Constraint in Mathematical Expression Recognition," in *Proceedings of the 15'th National Conference on Artificial Intelligence*. AAAI, 1998, pp. 784–791.

[3] S. Lavirotte, L. Pottier, and L. Pottier, "Mathematical Formula Recognition Using Graph Grammar," in *Proceedings of the SPIE*, 1998, pp. 44–52.

[4] J. Mas, J. Llados, G. Sanchez, and J. A. P. Jorge, "A syntactic approach based on distortion-tolerant adjacency grammars and a spatial-directed parser to interpret sketched diagrams," *Pattern Recogn.*, vol. 43, no. 12, pp. 4148–4164, Dec. 2010.

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

[6] Y. Qi, M. Szummer, and T. Minka, "Diagram structure recognition by Bayesian conditional random fields," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, (CVPR).*, vol. 2, june 2005, pp. 191 – 196 vol. 2.

[7] G. Feng, C. Viard-Gaudin, and Z. Sun, "On-line hand-drawn electric circuit diagram recognition using 2D dynamic programming," *Pattern Recogn.*, vol. 42, no. 12, pp. 3215–3223, Dec. 2009.

[8] T. Werner, "A Linear Programming Approach to Max-Sum Problem: A Review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 7, pp. 1165 –1179, july 2007.

[9] A.-M. Awal, G. Feng, H. Mouchere, and C. Viard-Gaudin, "First experiments on a new online handwritten flowchart database." in *DRR'11*, 2011, pp. 1–10.

[10] A. Lemaitre, H. Mouchére, J. Camillerapp, and B. Coüasnon, "Interest of syntactic knowledge for on-line flowchart recognition," in *Nineth IAPR International Workshop on Graphics Recognition, 2011. GREC 2011*, 2011, pp. 85–88.

[11] Z. Yuan, H. Pan, and L. Zhang, "Advances in Blended Learning." Berlin, Heidelberg: Springer-Verlag, 2009, ch. A Novel Pen-Based Flowchart Recognition System for Programming Teaching, pp. 55–64.

[12] H. Miyao and R. Maruyama, "On-line handwritten flowchart recognition, beautification, and editing system," in *ICFHR '12: Proceedings of the 13th International Conference on Frontiers in Handwriting Recognition*, 2012, pp. 83–88.

[13] S. Otte, D. Krechel, M. Liwicki, and A. Dengel, "Local feature based online mode detection with recurrent neural networks," in *ICFHR '12: Proceedings of the 13th International Conference on Frontiers in Handwriting Recognition*, 2012, pp. 531–535.

[14] E. Indermühle, V. Frinken, and H. Bunke, "Mode detection in online handwritten documents using BLSTM neural networks," in *ICFHR '12: Proceedings of the 13th International Conference on Frontiers in Handwriting Recognition*, 2012, pp. 302–307.

[15] M. Bresler, D. Průša, and V. Hlaváč, "Simultaneous Segmentation and Recognition of Graphical Symbols using a Composite Descriptor," in *CVWW '13: Proceedings of the 18th Computer Vision Winter Workshop*. Pattern Recognition and Image Processing Group, Vienna University of Technology, 2013, pp. 16–23.

[16] "Toulbar2 solver," http://carlit.toulouse.inra.fr/cgi-bin/awki.cgi/ToolBarIntro.

[17] "IBM ILOG CPLEX Optimizer," http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/.