Turbo recognition: a statistical approach to layout analysis

Taku A. Tokuyasu[a] and Philip A. Chou[b]

[a] Computer Science Division, University of California, Berkeley, CA
[b] Microsoft Research, Microsoft Corporation, Redmond, WA

ABSTRACT

Turbo recognition (TR) is a communication theory approach to the analysis of rectangular layouts, in the spirit of Document Image Decoding. The TR algorithm, inspired by turbo decoding, is based on a generative model of image production, in which two grammars are used simultaneously to describe structure in orthogonal (horizontal and vertical) directions. This enables TR to strictly embody non-local constraints that cannot be taken into account by local statistical methods. This basis in finite state grammars also allows TR to be quickly retargetable to new domains. We illustrate some of the capabilities of TR with two examples involving realistic images. While TR, like turbo decoding, is not guaranteed to recover the statistically optimal solution, we present an experiment that demonstrates its ability to produce optimal or near-optimal results on a simple yet nontrivial example, the recovery of a filled rectangle in the midst of noise. Unlike methods such as stochastic context free grammars and exhaustive search, which are often intractable beyond small images, turbo recognition scales linearly with image size, suggesting TR as an efficient yet near-optimal approach to statistical layout analysis.

Keywords: document image decoding layout analysis graphical grammar turbo ML MAP likelihood posteriori stochastic

1. INTRODUCTION

Turbo recognition[1] (TR) is a new approach to layout analysis in the tradition of Document Image Decoding[2] (DID). As in all DID-like work, it is characterized by a communication theory approach, with a focus on statistical optimality. Recognition as a general concept is defined as the recovery of a source message from the observed data.

This framework applies quite naturally in the context of optical character recognition[3] (OCR), for example, where a given text line can be viewed as a linear sequence of character templates (perhaps with vertical jitter). The horizontal coordinate along the text line plays the same role as time in applications such as speech recognition. Methods such as hidden Markov models (HMMs) can therefore be effectively applied.

Layout analysis is one of the basic tasks of document image analysis (DIA) for which the two-dimensional (2D) nature of the underlying signal (scanned images) is often an inherent part of the problem. Here the connection to the usual communication theory framework is less obvious. To deal with such problems, DID has traditionally turned to stochastic attribute grammars.[4] This general approach, akin to that used in some document authoring tools, allows the analysis of spatial constructions[5,6] considerably beyond the reach of simple one-dimensional HMMs. It has been successfully applied, for instance, to the recognition of mathematical formulas.[7] Previous studies[8] show, however, that the straightforward application of stochastic context-free grammars (SCFGs) can quickly become intractable for common layout analysis tasks, given that SCFG parsing (via, e.g., CYK methods) scales as $n^{5/2}$ in time and n^2 in space, where n is the number of pixels. The use of controlled heuristics that retain statistical optimality, introduced quite successfully in the OCR context,[9] may also be useful in the case of layout analysis, though this possibility has yet to be thoroughly explored.

Turbo recognition is a form of DID addressed especially to the recognition of rectangular layouts. It can be thought of as trading off some generality for speed, which is made possible by a unique factorization of the problem. The key idea in TR is to use *two* grammars to describe horizontal and vertical structure simultaneously. The decoding time then scales linearly in the number of pixels (times the number of turbo iterations). TR gains its inspiration from turbo codes,[10,11] a breakthrough development in communication theory that exhibit extremely low bit error rate in the presence of noise.

Previous applications of HMM-like models to two dimensions include pseudo-2D HMMs[12] and 2D HMMs[13] (a form of Markov field model). These applications focus on tasks such as keyword spotting and image classification.

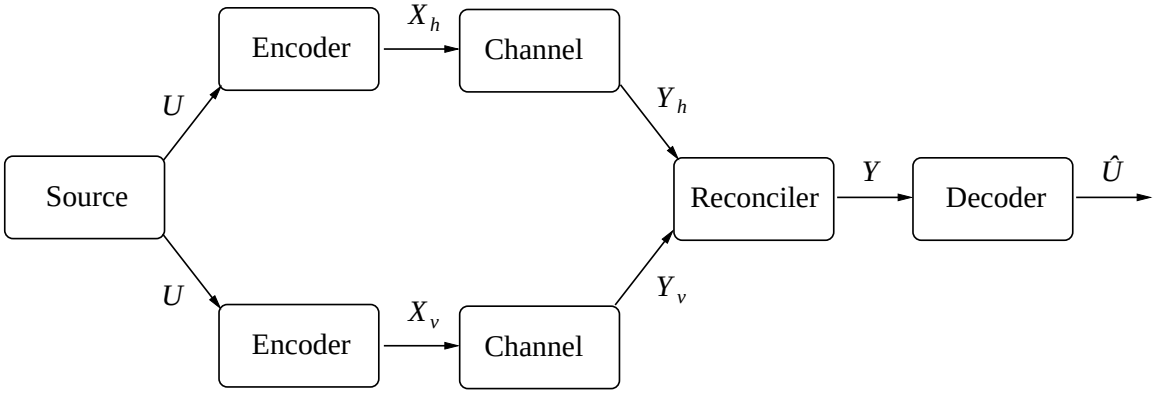


Figure 1. Communication system view of turbo recognition.

Turbo recognition, while similar in spirit to such methods, is distinguished in particular by its ability to embody non-local constraints. For instance, it is a simple matter to set up a TR grammar to describe a rectangular region, in particular the linear alignment of its edges. This concept cannot be captured exactly by methods that rely only on the statistical correlation of neighboring pixels. We will return to this point below.

2. TURBO RECOGNITION

The communication system view of turbo recognition is presented in Figure 1. This differs from the usual DID scheme in having two encoding channels. Here a single message U produced by the source is encoded into the two ideal images X_h and X_v (where the subscripts h and v stand for "horizontal" and "vertical," respectively). These are transmitted through separate channels, where they each get subjected to an independent noise process. This results in two corrupted images Y_h and Y_v. The reconciler passes these images to the decoder as a single image Y if an only if Y_h and Y_v are identical. Otherwise the reconciler passes a null image to the decoder. In practice, the decoder will never observe a null image. The reconciler is simply a mathematical artifice that allows us to construct a joint probability model for U, X_h, X_v, and Y.

We will call U the input field, X (generically) the output field, and Y the observed field. The 2D field U is the message we are trying to recover from the observed data Y. The minimum probability of error is achieved when the decoder returns the field $\hat{U}$ that maximizes the posterior, $\hat{U} = \arg\max_U P(U|Y)$.

Turbo recognition, being based on turbo decoding, is an iterative approach to solving (possibly approximately) * for the posterior $P(U|Y)$. Each turbo iteration consists of two passes, one applying the forward-backward[11] algorithm on the horizontal transducers, and the second applying forward-backward on the vertical transducers. This process is repeated to convergence of the posterior $P(U|Y)$. In our experience, three to five turbo iterations are sufficient. Details of the algorithm can be found in our DLIA paper.[1] Our current implementation of the algorithm is based on belief propagation in graphical models.[11,16]

Operationally, the primary task in defining a given TR decoder is to specify the grammars for the two encoders in Figure 1. These are finite-state transducers† (FSTs), which each take as input the 2D message field U and transduce it into a noise-free image X. The channel that relates X and Y also has to be specified. At present we do this manually, based roughly on the level of noise we expect to see in the image.

For example, the FST in Figure 2a accepts the input sequences described by the regular expression $A^+ \mid B^+C^+B^+$, while that in Figure 2b accepts the input sequences $A^+B^+A^+ \mid A^+C^+A^+$. Taken together, these two FSTs describe a layout such as the one shown in Figure 3, where each row is a sequence accepted by the first FST, and each column is accepted by the second. If we associate the input label C with an output symbol "1" (or "black"), and the remaining input symbols with the output symbol "0" (or "white"), we see that the noise-free image output by these FSTs is of a single black rectangle. Note that although A and B are both transduced to

*The status of turbo decoding with regard to statistical optimality is still unclear. Turbo codes (and hence turbo recognition) are an instance of a graphical model with loops, on which research is ongoing.[15]

†Non-finite state machines, such as recursive transition networks, could also be used, in principle.

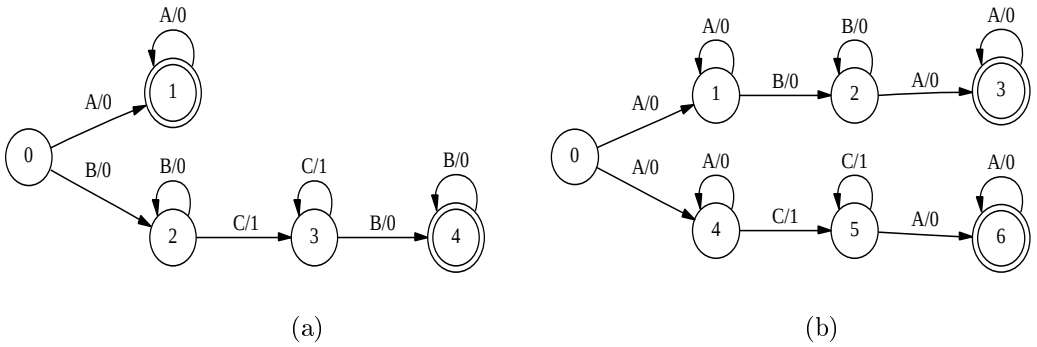


Figure 2. Finite state transducers: (a) horizontal; (b) vertical. The start state is labeled 0 in both instances, and accepting states are marked with double circles. Transitions are marked with an input and output label, in the format "input/output".

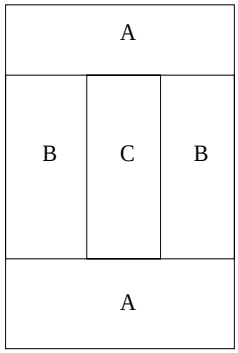


Figure 3. Sample layout described by the FSTs in Figure 2. All of the pixels in a given rectangular region are meant to be labeled by the symbol in its center.

0, both of them are required by the grammar, for otherwise the grammar would describe a general convex shape and not just rectangles.

These FSTs, together with the other aspects of the TR model, e.g. a prior distribution over U (which can be simply a constant) and a noise channel (e.g. bit-flip noise), define a probability model for the variables U, X and Y, for which we can then apply the turbo recognition algorithm.

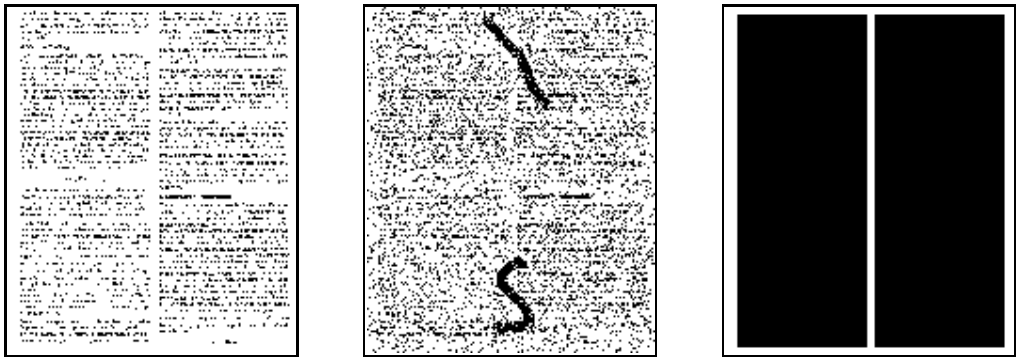
2.1. Example

As a more detailed illustration of the use of TR, consider the two-column text image shown in Figure 4a. The image dimensions are 153 by 184 pixels, corresponding to roughly 20 dpi. Figure 4b shows the original image with 10% bit-flip noise and pencil marks added.

We attempt to segment this latter image, in particular to extract the two text columns, with a grammar describing the following structure:

$$\begin{array}{ccccc} A & B & B & B & A \\ D & C & D & C & D \\ A & B & B & B & A \end{array}$$

Thus, the horizontal grammar is described by the regular expression $A^+B^+A^+|D^+C^+D^+C^+D^+$, and the vertical grammar by $A^+D^+A^+|B^+C^+B^+|B^+D^+B^+$. The symbol C represents a text block, while D is a "gutter" symbol, B represents header and footer regions of the original page, and A represents the corner regions. This summarizes the structure of the input field U. For the output field X, we use a three-symbol alphabet $\{0, 1, 2\}$, which gives us some flexibility in describing the differing print densities we expect to see on the page. We associate both A and B with output symbol 0, C with 1, and D with 2 (see Figure 5a). Finally, the alphabet for the observed field Y is $\{0, 1\}$, corresponding to white and black pixels, respectively. The channel describing the relation between X



(a) (b) (c)

Figure 4. (a) Original two-column text image; (b) Noisy test image; (c) TR two-column result.

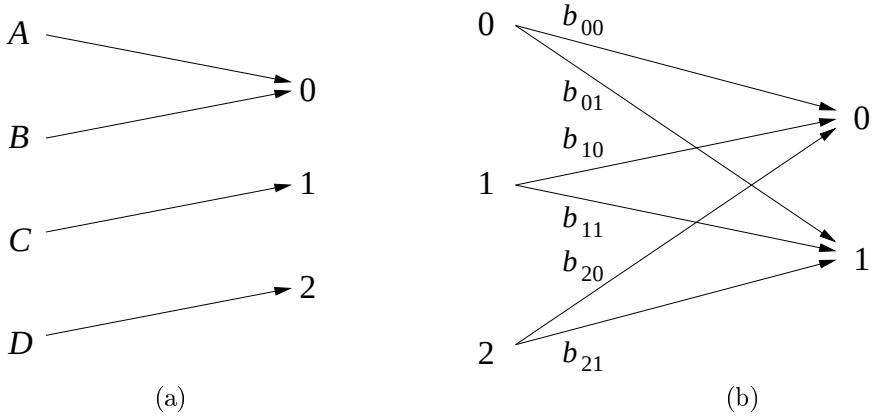


Figure 5. (a) Deterministic relation between U and X; (b) Noise channel between X and Y.

and Y is shown in Figure 5b, where the noise parameters b_{ij} are given by

$$\begin{array}{cc} 0.5 & 0.5 \\ 0.2 & 0.8 \\ 0.12 & 0.88 \end{array}$$

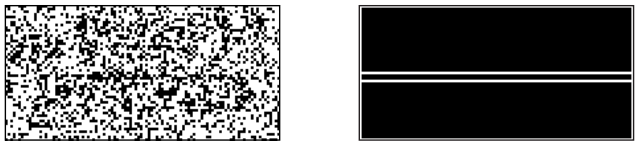
We found it convenient to use the "reverse-video" version of the test image, making it appropriate to use a channel where, for example, the output symbol 2 is quite likely to produce a black observed symbol.

Figure 4c is the result of TR, where the C symbol is printed in black. Three turbo iterations were used, which required 14 seconds on a 366 MHz Windows Pentium PC running Java. This simple example highlights the ability of TR to accumulate local evidence over long distances and disconnected regions, using a model (given by the grammar) that enforces non-local constraints. This distinguishes it from methods such as projection profiles and pure HMM methods.

Such a TR model can be quickly retargeted to deal with an example from an entirely different domain (Figure 6a). This shows a mathematical expression,

$$\frac{\pi a^{2\nu-1} z^{\nu-1/3}}{2^{2\nu+1} c^{\nu+1/2}}$$

with 40% bit-flip noise added. Retargeting TR is about as simple as writing down the appropriate regular expressions. In order to capture the concept of "numerator," "divide bar", and "denominator," we use



(a) (b)

Figure 6. a) A math fraction with 40% bit-flip noise; b) TR result.

$A^+ \mid B^+C^+B^+ \mid B^+D^+B^+$ for the horizontal grammar and $A^+B^+A^+B^+A^+B^+A^+ \mid A^+C^+A^+D^+A^+C^+A^+$ for the vertical grammar, which describes the following structure:

$$\begin{array}{ccc} A & A & A \\ B & C & B \\ A & A & A \\ B & D & B \\ A & A & A \\ B & C & B \\ A & A & A \end{array}$$

The channel parameters in this case were $\{\{0.9, 0.1\}, \{0.85, 0.15\}, \{0.05, 0.95\}\}$, which were arrived at with a modicum of manual tuning. The TR result is given in Figure 6b, where both C and D are printed in black. The divide bar is effectively extracted, with the numerator and denominator regions also defined above and below it. We can focus on the use of this grammar as simply a "divide bar finder," which utilizes knowledge of spatial structure to enhance its performance. Similar applications involving mathematical expressions, for example, are easily envisioned.

2.2. Experiment

We now turn to a study of the ability of TR to recover optimal solutions. As mentioned previously, TR is based on a graphical model with loops, and as such it is not yet clear in what contexts optimal recovery can be guaranteed, if at all. As a prototypical example of more general layout tasks, which is of interest in its own right, we consider the "one rectangle problem." That is, given a binary image of a single black rectangle on a white background, which has been corrupted by bit-flip noise with (pixel-wise independent) probability p, recover the original black rectangle. This can be interpreted as recovering the rectangle that maximizes the likelihood of the observed image. We investigate how well a one-rectangle turbo recognition model, such as the one described in the previous section, can solve this problem. When all U fields are equiprobable (as we are assuming), maximizing the posterior probability $P(U|Y)$ is equivalent to maximizing the likelihood $P(Y|U)$. The experiment thus involves two aspects, how well TR recovers the original black rectangle, and how well TR recovers the ML (optimal) solution.

The corresponding one-dimensional problem, that of finding the most likely single black interval given a noisy sequence of bits,[‡] can be solved easily via dynamic programming (DP), e.g. using HMM methods. However, to the authors' knowledge, no tractable DP methods exist in two dimensions.[§] For these experiments, we resort to an exhaustive search to recover the maximum likelihood solution. This method scales as n^2, where n is the number of pixels. Turbo recognition scales linearly in n, times the number of turbo iterations. Turbo recognition thus may be the first method that can efficiently solve this problem.

It turns out that we must soften this claim somewhat, as TR does not always recover the optimal solution. We obtain the best results by applying the turbo iterative updates in a graduated fashion, inspired by the use of deterministic annealing for HMM design.[14] For instance, on 1000 noisy images, derived from a 12 by 11 black rectangle corrupted with bit-flip noise at $p = 0.2$ (see Figure 7), TR yields the optimal solution 97% of the time. When we count all TR solutions whose corner coordinates are within ± 1 pixel of the corresponding ML ones, this

[‡]By convention, we can say 1 is equivalent to "black".

[§]The problem can be treated as a one dimensional problem by considering an entire row of pixels to be a single state. This scales exponentially in the number of pixels and quickly becomes intractable on anything besides tiny images.

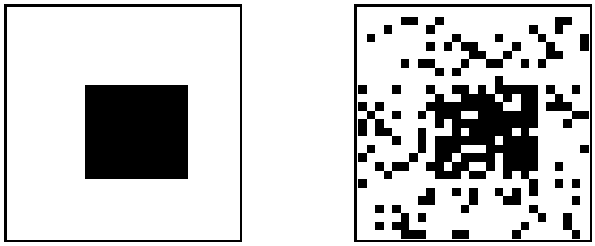


Figure 7. Black rectangle and a typical noisy version at $p = 0.2$.

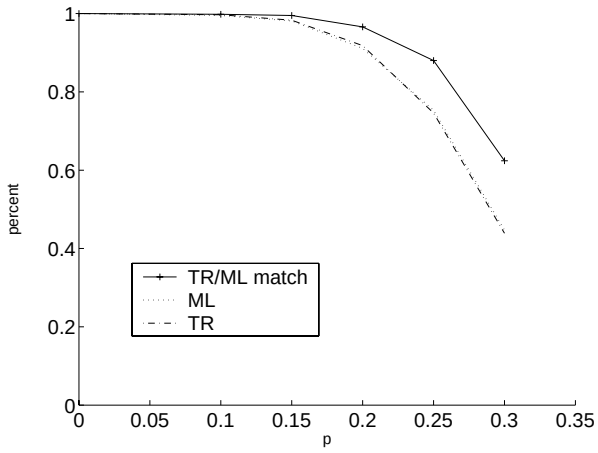


Figure 8. Recognition results for the one-rectangle problem. The solid line describes the probabilities with which the TR solutions match the ML solutions. The dotted and dash-dotted lines describe the probabilities with which the ML and TR solutions, respectively, match the original clean image.

number rises to 99.6%, and all TR solution are within a two pixel tolerance of the ML rectangles. To provide some context for these results, we note that at this noise level ($p = 0.2$), even the optimal ML solution recovers the original clean image only 91% of the time.

The results for a range of noise levels from $p = 0$ to 0.3 are summarized in Figure 8. These curves describe the probabilities that the TR solution matches the ML solution, and that each of these match the original image (the latter two curves are difficult to distinguish at this resolution). We see that TR and ML are essentially equivalent in their ability to recover the original image. Also, the match between TR and ML begins to degrade at about the same point that ML itself starts failing to recover the original image.

Future work includes analyzing the cases where TR and ML fail to match. Recent work by Freeman and Weiss,[15] for instance, shows that maxproduct message passing (*aka* belief revision[16]) in loopy graphical models such as TR produces solutions that are optimal over an extended neighborhood of that solution.

For this small experimental image, TR is about twice as fast as ML exhaustive search, taking about 0.8 seconds per image (with five turbo iterations each). The difference in complexity scaling rapidly makes the speed differential more significant when applied to larger or more complex images.

3. CONCLUSION

The above results are encouraging, since they suggest TR can provide near-optimal solutions to layout analysis problems. TR thus holds promise as a principled approach to using stochastic grammars in two dimensions. Using low resolution (e.g. 20 dpi) scanned document images, our preliminary experiments show that TR can effectively recover two-column formats, four-column formats, one block on top of two columns (i.e., a rudimentary "journal article title page" grammar), etc. These experiments suggest further innovations to extend the present formalism, e.g. incorporating OCR into the framework.

The problem of specifying "groundtruth", especially for higher level notions like document layout, is a continuing problem, which for instance makes it difficult to compare research results. We note that, in the case of TR (and all DID-like approaches), the question of what constitutes groundtruth is simple: it is the original message that the decoding algorithm is attempting to recover. To the extent that such an approach can be applied to a given problem, this provides a solid basis for evaluation.

4. ACKNOWLEDGEMENTS

TAT thanks Yair Weiss and Richard Fateman for conversations on this work, and the support of the Berkeley Digital Library Project under NSF grant number CA98-17353.

REFERENCES

1. T.A. Tokuyasu and P.A. Chou. An iterative decoding approach to document image analysis. Document Layout Interpretation and its Applications (DLIA99) September 18 , 1999, Bangalore, India. http://www.science.uva.nl/events/dlia99/final_papers/chou.pdf
2. G. E. Kopec and P. A. Chou. Document image decoding using Markov sources. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 16(6):602–617, June 1994.
3. G. E. Kopec. Document image decoding in the UC Berkeley digital library. *Document Recognition III*, Proc. of the SPIE, 1996, vol.2660:2-13.
4. P.A. Chou and G.E. Kopec. A stochastic attribute grammar model of document production and its use in document image decoding. *Document Recognition II*, Proc. of the SPIE, 2422:66-73, 1995.
5. P. A. Chou. Recognition of equations using a two-dimensional stochastic context-free grammar. In *Visual Communications and Image Processing*, pages 852–863, Philadelphia, PA, November 1989. SPIE.
6. M. Tomita Parsing 2-dimensional language. In *International Workshop on Parsing Technologies*. Carnegie Mellon Univ, p. 414-24, 1989.
7. J. F. Hull. Recognition of mathematics using a two-dimensional trainable context-free grammar. Master's thesis, MIT, Cambridge, MA, June 1996.
8. T. A. Tokuyasu. Unpublished.
9. A.C. Kam and G.E. Kopec. Document image decoding by heuristic search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(9):945-50, Sept. 1996.
10. C. Berrou, A. Glavieux, and P. Thitimajshima. Near Shannon limit error-correcting coding and decoding: Turbo codes. 1. In *Proc. of ICC' 93*, IEEE, vol.2:1064-70, 1993.
11. Brendan J. Frey. *Graphical models for machine learning and digital communication* The MIT Press, Cambridge, MA, c1998.
12. O.E. Agazzi and S.-S. Kuo. Pseudo two-dimensional hidden Markov models for document recognition. *AT&T Technical Journal*, 72(5):60–72, Sept-Oct, 1993.
13. Jia Li, A. Najmi, and R.M. Gray. Image classification by a two-dimensional hidden Markov model. *IEEE Transactions on Signal Processing*, 48(2):517–33, Feb. 2000.
14. D. Miller, K. Rose, and P.A. Chou. Deterministic annealing for trellis quantizer and HMM design using Baum-Welch re-estimation. In *Proc. of ICASSP '94*, IEEE, vol.5:V/261-4, 1994.
15. W.T. Freeman and Y. Weiss. On the fixed points of the max-product algorithm. MERL TR99-39. submitted to IEEE Transactions on Information Theory.
16. J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, San Mateo, CA, 1988.