

EMMCVPR 2011, St. Petersburg

Minimizing Count-based High Order Terms in Markov Random Fields

Thomas Schoenemann

Center for Mathematical Sciences
Lund University, Sweden

Abstract. We present a technique to handle computer vision problems inducing models with very high order terms - in fact terms of maximal order. Here we consider terms where the cost function depends only on the *number* of variables that are assigned a certain label, but where the dependence is arbitrary.

Applications include image segmentation with a histogram-based data term [28] and the recently introduced marginal probability fields [31]. The presented technique makes use of linear and integer linear programming. We include a set of customized cuts to strengthen the formulations.

1 Introduction

When global methods for a certain kind of optimization problems with binary terms became known [13], for several years research in computer vision focused on such problems, and they are very well understood today.

Over the past few years the trend has moved more and more towards higher order terms which are generally more difficult to solve but also provide better models [22,30,20,24,17].

Research on optimizing such models is split into two fields: methods designed for terms of fairly low order, e.g. up to five, and methods that address very high order terms up to the maximal order of the number of variables.

The first class includes the works [21,29,33,30,20] and their complexity grows exponentially with the order of the term. Some of them [30,20] do however admit efficient solutions for specific high order problems. There are no restrictions on the form of the terms, but performance can of course differ greatly - already problems with binary terms are in general NP-hard.

The methods in this first group are split into two subclasses: some of them [29,30,20], usually message passing algorithms, are directly based on the higher order terms. Alternatively, there are methods that first convert higher order terms into unary and binary terms [5,32] and then make use of appropriate inference techniques [32,14,2]. In special cases even the global minimum can be computed [3,11].

The second class of methods handles very high order terms, and they need to make assumptions on the form of the terms. The method [18] handles any concave dependence on the number of variables that are assigned a certain label. For binary labeling problems such terms are optimized globally (when all binary terms are submodular), for multi-label problems move-based algorithms are used. An alternative method for a largely overlapping class of models was recently given in [10]. Furthermore, general submodular functions (possibly with high order terms) can be optimized in polynomial time [26,15], but to our knowledge these algorithms have never been tested in a computer vision context.

Moreover, for a broad class of problems (including all those discussed in this paper) message passing approaches can be implemented with a polynomial dependence on the maximal order of the high order terms [23,27]. Another specialized solution for a large part of this class was given in [17], but tested only on small synthetic problems.

Finally, there are some problem-specific solutions based on dual decomposition [28,31]. As the respective problems will be addressed in this paper, we review them more closely below. Before, however, we give a brief summary of this paper.

This work: We address the class of very high order terms where the cost depends only on the number of variables that are assigned a label, but without any restrictions on the form of this dependence. Our method is based on (suboptimally) solving integer linear programs (ILPs), where we make use of standard packages. We first solve the linear programming relaxations, then apply a branch-and-cut scheme where we employ the set of cuts we derived in [25].

Note that our ILPs are different from those considered in the above cited works, in particular from the standard max-product message passing techniques: for terms of maximal order these latter produce exponentially large formulations that in some special cases can be handled implicitly. In contrast, our formulation is very compact, so we can make use of standard integer linear programming solvers.

Likewise, our strategy after solving the linear programming relaxation differs greatly from the cutting planes scheme described in [30] which introduces extra variables. We only introduce extra constraints. We demonstrate applications for the following problems:

Histogram Image Segmentation. A popular model for image segmentation (e.g. [4,28]) combines a length-based regularity term with appearance terms based on the log-probabilities of fairly general distributions (e.g. histograms or Gaussian Mixtures). These distributions are themselves unknown and to be adapted to the given image. Traditionally, these models have been addressed via alternating minimization (AM). For histograms, recently [28] cast the problem as finding the global optimum of a labeling problem with terms of maximal order. Their solution is based on dual decomposition. However, we found that this approach only works if seed nodes are given. We give a solution that works in the fully

unsupervised setting and also extends to multi-label problems (however, here it does not beat AM).

Marginal Probability Fields. It was recently proposed [31] to extend the traditional Markov Random Field framework by high order terms measuring how well a labeling reflects a-priori known marginal statistics of certain features. The authors give a dual-decomposition optimization scheme. However, in contrast to our work they neglect consistency: the method assigns labels independently to single nodes and pairs of nodes, although the latter are already defined by the former. Moreover, while they only explored unary and pairwise terms we will also include triple constellations.

2 Count-based Terms as Integer Linear Programs

We start with a description of the general class of labeling problems we consider and simultaneously indicate applications for computer vision.

In this paper we consider the problem of assigning each of the nodes p in a certain finite set $\mathcal{V} \subseteq \mathbb{R}^2$ – typically the pixels in a given image – a label $y_p \in \mathcal{L} = \{1, \dots, K\}$, with K a given constant. We are interested in finding the *best* labeling, where “best” is defined as the minimum of an energy function with unary, pairwise and a certain kind of higher order terms. These last terms depend on all nodes simultaneously and their cost are a function of the number of times a certain labeling constellation is observed.

In the simplest case these constellations are the labels of single pixels and the problem is of the form

$$\begin{aligned} \min_{\mathbf{y}} \sum_{p \in \mathcal{V}} D_p(y_p) + \sum_{(p,q) \in \mathcal{N}} V_{p,q}(y_p, y_q) \\ + \sum_{l=1}^K f_l \left(\sum_{p \in \mathcal{V}} \delta(y_p, l) \right) \quad , \end{aligned} \quad (1)$$

where \mathcal{N} is a neighborhood system and $\delta(\cdot, \cdot)$ the Kronecker- δ , i.e. 1 if both its arguments are equal, otherwise 0. The real-valued functions $D_p(\cdot)$, $V_{p,q}(\cdot, \cdot)$ and $f_l(\cdot)$ can be chosen freely - there are no restrictions on them.

In a slightly more general form, we can have several higher order functions per label¹. Instead of a single function $f_l : \{0, \dots, |\mathcal{V}|\} \rightarrow \mathbb{R}$ we now allow N_l functions where each of them can collect counts over its own subset $\mathcal{S}_l^i \subseteq \mathcal{V}$ of nodes. The model then reads

$$\begin{aligned} \min_{\mathbf{y}} \sum_{p \in \mathcal{V}} D_p(y_p) + \sum_{(p,q) \in \mathcal{N}} V_{p,q}(y_p, y_q) \\ + \sum_{l=1}^K \sum_{i=1}^{N_l} f_l^i \left(\sum_{p \in \mathcal{S}_l^i} \delta(y_p, l) \right) \quad . \end{aligned} \quad (2)$$

¹ One can also collect counts across different labels, but we will not explore this here.

Example. The class (2) contains the problem of histogram-based image segmentation when stated as a purely combinatorial problem [28]. Given an image $I : \mathcal{V} \rightarrow \{0, \dots, 255\}$, the problem is to minimize

$$\sum_{p \in \mathcal{V}} -\log[P_{y_p}(I(p))] + \sum_{(p,q) \in \mathcal{N}} \frac{\lambda}{\|p - q\|} (1 - \delta(y_p, y_q)) \quad (3)$$

with respect to both the probability distributions $P_1(\cdot), \dots, P_K(\cdot)$ and the labels y_p for $p \in \mathcal{V}$. For a given labeling \mathbf{y} the minimizing distributions are given by

$$P_l(k) = \frac{\sum_{p: I(p)=k} \delta(y_p, l)}{\sum_{p \in \mathcal{V}} \delta(y_p, l)} \quad , k \in \{0, \dots, 255\}$$

(where $0/0$ is defined as 0). The key observation of [28] is that when inserting the negative logarithm of this term into the above functional one obtains a purely combinatorial problem of the form (2). The unary terms in (3) can now be grouped together into higher order terms, and defining $h(n) = n \log(n)$ (with $h(0) = 0$) the problem is written as

$$\begin{aligned} & \sum_l h\left(\sum_{p \in \mathcal{V}} \delta(y_p, l)\right) + \sum_l \sum_{k=0}^{255} -h\left(\sum_{p: I(p)=k} \delta(y_p, l)\right) \\ & + \sum_{(p,q) \in \mathcal{N}} \frac{\lambda}{\|p - q\|} (1 - \delta(y_p, y_q)) \quad . \end{aligned}$$

Very similar derivations can be given for color images and histograms with arbitrarily defined bins.

2.1 Integer Programming Formulation

We now show how the above energy minimization problems can be cast as integer linear programs, i.e. as minimizing a linear cost function subject to linear constraints and integrality conditions on the variables.

We use the common concept of binary indicator variables $x_p^l \in \{0, 1\}$ where $x_p^l = 1$ indicates that $y_p = l$. With these variables, the unary terms are readily expressed as a linear cost function. To write the binary terms in a linear way, we consider variables $x_{p,q}^{l_1, l_2} \in \{0, 1\}$ that we want to be 1 if and only if $y_p = l_1$ and $y_q = l_2$. All these binary variables are grouped into a vector \mathbf{x} and the associated linear cost is denoted $\mathbf{c}_x^T \mathbf{x}$, where \mathbf{c}_x has elements

$$c_p^l = D_p(l) \quad \text{and} \quad c_{p,q}^{l_1, l_2} = V_{p,q}(l_1, l_2) \quad .$$

To express the count-based terms we introduce variables $z_{l,i}^n \in \{0, 1\}$ (for $n \in \{0, \dots, |\mathcal{S}_l^i|\}$) that we want to be 1 if and only if the count for the function f_l^i is equal to n . All these variables are grouped into a vector \mathbf{z} and the associated cost $\mathbf{c}_z^T \mathbf{z}$ has elements $c_{l,i}^n = f_l^i(n)$. In addition, there are three sets of constraints to

be satisfied, where the first states that every node must have exactly one label as well as that for all l and i exactly one of the count variables $z_{l,i}^n$ must be 1. The second states that the binary variables must be consistent with the values induced by the unary ones. As these are quite standard constraints that arise in many message passing approaches, we defer the equations to (5) below.

In contrast, the last set of constraints is not at all common for computer vision. It states that the count variables need to be consistent with the associated node variables:

$$\sum_{p \in S_i^l} x_p^l = \sum_{n=0}^{|S_i^l|} n \cdot z_{l,i}^n \quad (4)$$

Together this results in the following integer linear program:

$$\begin{aligned} & \min_{\mathbf{x}, \mathbf{z}} \mathbf{c}_x^T \mathbf{x} + \mathbf{c}_z^T \mathbf{z} \\ & \text{s.t.} \quad \sum_l x_p^l = 1 \quad \forall p \in \mathcal{V} \\ & \quad \sum_{n=0}^{|S_i^l|} z_{l,i}^n = 1 \quad \forall l \in \mathcal{L}, i = 1, \dots, N_l \\ & \quad x_p^l = \sum_{q \in \mathcal{N}(p)} \sum_{l'} x_{p,q}^{l,l'} \quad \forall p \in \mathcal{V}, l \in \mathcal{L} \\ & \quad \sum_{p \in S_i^l} x_p^l = \sum_{n=0}^{|S_i^l|} n \cdot z_{l,i}^n \quad \forall l \in \mathcal{L}, i = 1, \dots, N_l \\ & \quad x_p^l \in \{0, 1\}, x_{p,q}^{l_1, l_2} \in \{0, 1\}, z_{l,i}^n \in \{0, 1\} \end{aligned} \quad (5)$$

Solving this kind of problem is in general NP-hard [28].

2.2 Special Cases

For a number of problems the ILP (5) can be written more compactly. In particular, for binary labeling problems one can replace all occurrences of $x_p^{l=2}$ by $1 - x_p^{l=1}$ and drop the constraints in the second line of (5).

Furthermore, for many regularity terms $V_{p,q}(l_1, l_2)$ one can reduce the number of pairwise variables and constraints. This includes the Potts model, where it is well-known [16,34] that

$$\begin{aligned} V_{p,q}(l_1, l_2) &= \lambda(1 - \delta(l_1, l_2)) \\ &= \frac{\lambda}{2} \sum_{l \in \mathcal{L}} |x_p^l - x_q^l| \quad , \end{aligned}$$

where $\lambda > 0$ is a smoothness weight. Now, it is well-known (e.g. [9]) that the absolutes in this expression can be written as linear programs:

$$\begin{aligned} \min_{\mathbf{x} \geq \mathbf{0}, \mathbf{a}_{\pm} \geq \mathbf{0}} \quad & \sum_{(p,q) \in \mathcal{N}} \sum_{l \in \mathcal{L}} \frac{\lambda}{2} [a_{p,q,l}^+ + a_{p,q,l}^-] \\ \text{s.t.} \quad & x_p^l - x_q^l = a_{p,q,l}^+ - a_{p,q,l}^- \end{aligned}$$

Such constructions significantly reduce the number of required variables. Finally, if a function $f_l : \{0, \dots, |\mathcal{S}_i^l|\} \rightarrow \mathbb{R}$ is *convex*, we can bypass the consistency constraints (4): such a function can be implemented as a set of inequalities, see section 4.2 and [17].

In this work we always make use of such reductions in the problem size.

2.3 Composite Features

So far the higher order terms depended on the number of *nodes* that were assigned a certain label. In a more general setting one can count any kind of features, e.g. by looking at certain pairs of nodes or certain triplet constellations.

Handling pairs of nodes is particularly easy since (5) already contains variables that explicitly reflect pairwise constellations. We only have to make sure that all relevant pairs are contained in the neighborhood system \mathcal{N} and slightly modify (4) so that the right hand side now contains the pairwise variables. In the case where the dependence is on the number of pairwise constellations (p, q) in the neighborhood system with the labels $y_p = l_1, y_q = l_2$, this reads

$$\sum_{(p,q) \in \mathcal{N}} x_{p,q}^{l_1, l_2} = \sum_{n=0}^{|\mathcal{N}|} n \cdot z_{l_1, l_2}^n$$

Similarly, one can introduce variables $x_{p,q,r}^{l_1, l_2, l_3}$ that express the constellations of triplets of nodes (see e.g. [30]). One then needs to introduce the corresponding consistency constraints between the node variables and the triplet variables, and modify the above count constraints so that they sum over the new variables.

3 Optimization Strategies

A number of useful integer linear programs were presented, and we now turn to the question of how to solve them, at least approximately. Here we make use of a combination of standard integer linear programming solvers (both open source and commercial) and specialized computer vision code. The latter is integrated as plug-ins into the solvers.

3.1 Linear Programming

Standard approaches to integer linear programming start with solving the associated linear programming relaxation – the linear program that arises when dropping the integrality constraints on the variables. We adopt this scheme, relying on the standard packages.

There are currently two classes of algorithms to solve linear programs. The first class is the class of *simplex algorithms* that find so-called *basic feasible* solutions. This is a prerequisite for most standard implementations of the so-called cutting planes method for integer programming. These algorithms do not have a polynomial time guarantee, but in practice they are often very efficient and memory saving. Moreover, there are very good open source implementations, e.g. Clp².

On the other hand, there are *interior point methods* that employ a Newton optimization scheme and come with a polynomial time guarantee. To solve the arising linear equation systems the sparse Cholesky decomposition is used. As this involves a lot of expertise it is usually best to rely on commercial products here. These products are often faster than the simplex method, but in our experience they require more memory, up to a factor of two. Also, they generally do not give basic feasible solutions, so one has to run a procedure called *crossover* afterwards.

We found that both methods can be useful, depending on what problem is handled.

3.2 Customized Cutting Planes

Linear programming relaxations often provide reasonably good lower bounds on the integer problem. However, it is hard to convert them into good integral solutions - a simple thresholding often performs very poorly.

Hence, linear programming is only the starting point of our method. Subsequently we apply two techniques. The first one is called *cut generation*, where a cut is nothing else than a linear inequality that is valid for all feasible integral solutions. One is interested in finding cuts that improve the current relaxation, i.e. its fractional optimum becomes infeasible when the cuts are added to the system. One says that such cuts are *violated* by the current relaxation.

Many approaches for cut generation are known and implemented in the standard solvers, where the most generally applicable method are probably the Gomory cuts [12]. In our setting we use a specialized class of cuts we presented in [25] and which allows to find violated cuts very efficiently. These cuts address consistency constraints of the form

$$\sum_{i=1}^I x_i = \sum_{n=0}^I n \cdot z_n$$

² <http://www.coin-or.org/projects/Clp.xml>

together with the constraints $\sum_{n=0}^I z_n = 1$, $\mathbf{z} \geq \mathbf{0}$. To motivate the cuts, we give a fractional solution for the case $I = 10$ and where we know that $\sum_{i=1}^{10} x_i = 3$. Then $z_0 = 7/10$, $z_{10} = 3/10$ and $z_i = 0$ for all other i is a feasible solution.

This will indeed be the optimal solution if the represented count-cost $f(\cdot)$ is *concave* in n . Note that this includes the function $-h(n)$ that was introduced for histogram-based image segmentation in Section 2.

The cuts are based on the following reasoning: if we know that all variables of a subset $C \subseteq \{x_1, \dots, x_I\}$ of size $|C| = N$ are 1, we can conclude that the count variables z_n for $n < N$ must all be 0. This can be expressed as the inequality:

$$\sum_{i:x_i \in C} x_i + \sum_{n=0}^{|C|-1} z_n \leq |C| \quad .$$

Violated cuts are efficiently found by sorting the variables. There are exponentially many sets C , but in practice sufficiently few corresponding cuts are violated. There is a closely related set of cuts, derived from the fact that whenever all variables in $C \subseteq \{x_1, \dots, x_I\}$ are 0, then the count variables z_n for $n > N$ must be 0:

$$- \sum_{i:x_i \in C} x_i + \sum_{n=|C|+1}^I z_n \leq 0$$

The derived classes of cuts are not sufficient to arrive at an integral solution. In fact, they are only useful for cost functions f_l^i that have regions of concavities. Even then, we found that for the original linear programming relaxation usually none of these cuts is violated – the respective fractional solutions usually set the non-count variables to values near 0.5. Here, even the standard cut generation methods produce either no cuts or quite dense ones (with 300 or more non-zero coefficients), which soon exhausts the available memory.

As a consequence, we combine the derivation of cuts with the second technique, branch-and-bound, into a *branch-and-cut* scheme.

3.3 Branch-and-Cut

Branch-and-cut is based on the method of branch-and-bound (e.g. [1]): the problem is hierarchically partitioned into smaller sub-problems, called *nodes*. At each node, the arising linear programming relaxation is solved, which gives a lower bound on the sub-problem. If the obtained solution is integral or the lower bound exceeds some known upper bound on the original problem, the node can be closed. This process will eventually find the optimum, but this may take exponential time.

All used solvers allow two kinds of interaction in this scheme (usually via so-called `callback`-functions): firstly, once the relaxation of a node has been solved, the user can provide his/her own routine to generate cuts - we use the cuts stated above. Secondly, we can provide a routine which generates an integral

solution from the fractional solution of the node. Here, for histogram-based image segmentation we include the well-known alternating minimization scheme with the help of graph cuts [6] and expansion moves [7], where the fractional solution serves as an initialization of the probabilities. We found this to produce much better solutions than the standard heuristics included in the solvers.

4 Experiments

The addressed class of models allows a great variety of applications, and here we consider three of them. We experimented with the commercial solver Gurobi and the open source solver CBC³, the results below were produced with Gurobi. All experiments were run on a 2.4 GHz Core2 Duo with 3 GB memory.

4.1 Histogram-based Image Segmentation

We start with histogram-based image segmentation as described in example 1 above. In this case, some of the high order terms are convex, the others are concave. Since the convex functions are strictly convex we cannot reduce the size of the ILP by including inequalities.

For the concave functions we tried the cut generation plug-in we described in section 3.2, but found it only mildly helpful. Hence, we include it for binary problems but (since it slows down the solver) we do not use it for multi-label problems. Further, since the problem is symmetric we strengthen the relaxation by assigning one of the pixels to region 0.

In addition, we provide a plugin to generate integral solutions from fractional ones, where we use an alternating minimization (AM) scheme, updating first the probability distributions, then the segmentation via graph cuts or expansion moves. We found that the produced integral solutions are of much higher quality than those produced by the standard methods of Gurobi (or other toolkits).

Further, we run the AM scheme a priori with two different initial segmentations, given by horizontally and vertically dividing the image into K parts, where K is the number of regions. The two resulting energies usually differ significantly and we take the lower one. This solution is then passed to the solver and serves as an initial upper bound.

Note that we deal with a fully unsupervised scenario, i.e. there are no seed nodes. We found that the recent work of [28] is not applicable here: parametric maxflow gives only two trivial solutions. This approach is closely related to a linear relaxation, and we found that our LP-relaxation alone is equally useless: its energy forms a reasonable lower bound, but most of the segmentation variables are set to (roughly) 0.5. This is useless for thresholding schemes.

Results. Finding the global optimum is illusory in practice, so we set a time limit of 2.5 hours. We ran our method on all 100 images of the test set of the Berkeley image database, downscaled to a resolution of 120×80 . In 75 cases our method

³ <http://www.coin-or.org/projects/Cbc.xml>

was able to find a lower energy solution than the starting point (the better of two runs of AM).

Figure 1 shows the cases with the most significant differences. Clearly AM tends to find solutions with a short boundary, and stays often close to its initialization. Our solutions are of lower energy and frequently more interesting as they often locate animals. Figure 2 shows images where our approach does not improve the initialization. For some of them this may well be the global optimum.

We experimented with 3-region segmentation, but since no better solution than the initial one was found we omit the images.

4.2 Binary Texture Denoising

Our next application is binary texture denoising as addressed in [8], [19]. Here one first runs a training stage on a given binary texture image and extracts certain relevant probability distributions. For example, one can collect the statistics of certain pairwise constellations, i.e. look at all pairs of pixels where the second pixel is obtained by shifting the first by a fixed displacement. One obtains distributions of the form

$$p_{\mathbf{t}}(l_1, l_2) = p\left(I(\mathbf{x}) = l_1, I(\mathbf{x} + \mathbf{t}) = l_2\right).$$

Now one wants to select a set of translation vectors \mathbf{t} that characterize the given texture rather than describe a random distribution. It is well-known that this is reflected in minimal entropies:

$$\sum_{l_1 \in \{0,1\}, l_2 \in \{0,1\}} p_{\mathbf{t}}(l_1, l_2) \log\left(p_{\mathbf{t}}(l_1, l_2)\right),$$

so we take the 15 translations with minimal entropies. We also tried adding the 7 most informative triple constellations, selected in the same way.

Given is now a noisy gray-value image, and we look for a binarized denoised image with data terms like in [8] and V-kernels to penalize the marginal statistics [31]. The (convex) V-kernels are easily expressed in terms of two inequalities, which reduces the size of the system a little. Since the linear programming relaxation is quite strong we only solve 10 nodes in the branch-and-cut subsequently.

Results Figure 3 confirms that indeed marginal probability fields [31] improve on the widely used Markov Random fields. Moreover, the Gurobi solver found the global optimum in no more than 8 minutes, suggesting that the problem may be rather easy to solve in many situations. We trained on the right part of the well-known Brodatz texture D101, then process a noisy version of a crop of the left part.

4.3 Completion of Binary Textures

In a related setting we are given a partial texture and want to inpaint the missing part. Figure 4 demonstrates that now MRFs and MPFs perform very differently:



Fig. 1. Joint histogram segmentation on images of size 120×80 . Left image of the pair: Alternating Minimization. Right: Our method.

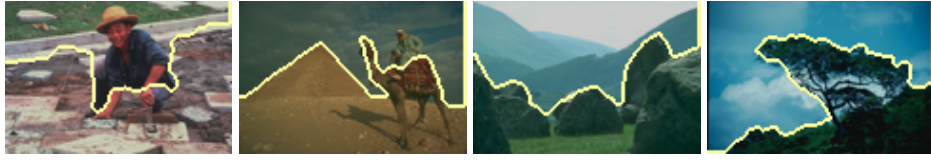


Fig. 2. Images where both approaches find the same result.



Fig. 3. Image denoising with MRFs and MPFs. We manually selected the best weighting parameter for each method. Both results are globally optimal.

the Markov Random Fields choose a constant fill value, whereas the MPF tries to respect the marginal statistics.

The linear programming relaxations are still quite strong and we let them follow by 10 nodes of branch-and-cut. This time we also test ternary terms (without branch-and-cut), but it can be seen that this does not result in performance gains (and it takes much longer). The running times are 1.5 hours for binary terms and roughly 10 hours for ternary terms.

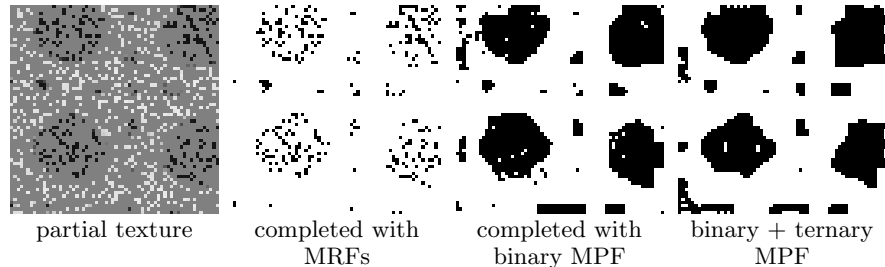


Fig. 4. Completion of partial textures (gray values indicate unknown regions) via MPFs and MRFs (via graph cuts).

Conclusion

We have proposed an integer linear programming framework to solve minimization problems with very high order terms depending on counts. For histogram-

based image segmentation it was shown that this improves over existing alternating minimization schemes, which again are very good plug-ins for standard solvers.

We furthermore showed that the recently introduced marginal probability fields can be handled and examined ternary terms. It was clearly demonstrated that in some cases this is a much more sensible approach than standard Markov Random Fields.

In future work we want to explore customized strategies to solve the arising linear programs.

Acknowledgements. We thank Fredrik Kahl for helpful discussions. This work was funded by the European Research Council (GlobalVision grant no. 209480).

References

1. T. Achterberg. *Constraint Integer Programming*. PhD thesis, Zuse Institut, TU Berlin, Germany, July 2007. 8
2. A. Ali, A. Farag, and G. Gimel'farb. Optimizing binary MRF with higher order cliques. In *European Conference on Computer Vision (ECCV)*, Marseille, France, Oct. 2008. 1
3. A. Billionet and M. Minoux. Maximizing a supermodular pseudo-boolean function: A polynomial algorithm for supermodular cubic functions. *Discrete Applied Mathematics*, 12(1):1–11, 1985. 1
4. A. Blake, C. Rother, M. Brown, P. Perez, and P. Torr. Interactive image segmentation using an adaptive GMMRF model. In *European Conference on Computer Vision (ECCV)*, Prague, Czech Republic, May 2004. 2
5. E. Boros and P. Hammer. Pseudo-Boolean optimization. *Discrete Applied Mathematics*, 123(1–3):155–225, Nov. 2002. 1
6. Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 26(9):1124–1137, 2004. 9
7. Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 23(11):1222–1239, 2001. 9
8. D. Cremers and L. Grady. Learning statistical priors for efficient combinatorial optimization via graph cuts. In *European Conference on Computer Vision (ECCV)*, Graz, Austria, May 2006. 10
9. G. Dantzig and M. Thapa. *Linear Programming 1: Introduction*. Springer Series in Operations Research. Springer, 1997. 6
10. A. DeLong, A. Osokin, H. Isack, and Y. Boykov. Fast approximate energy minimization with label cost. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, San Francisco, California, June 2010. 2
11. D. Freedman and P. Drineas. Energy minimization via graph cuts: Settling what is possible. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, San Diego, California, June 2005. 1
12. R. Gomory. Outline of an algorithm for integer solutions to linear programs. *Bulletin of the American Mathematical Society*, 64:275–278, 1958. 7
13. D. Greig, B. Porteous, and A. Seheult. Exact maximum *a posteriori* estimation for binary images. *Journal of the Royal Statistical Society, Series B*, 51(2):271–279, 1989. 1

14. H. Ishikawa. Transformation of general binary MRF minimization to the first order case. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2010. To appear. [1](#)
15. S. Iwata. A fully combinatorial algorithm for submodular function minimization. *Journal of Combinatorial Theory Series B*, 84(2):203–212, Mar. 2002. [2](#)
16. J. Kleinberg and E. Tardos. Approximation algorithms for classification problems with pairwise relationships: metric labeling and Markov Random Fields. In *Symposium on Foundations of Computer Science*, 1999. [5](#)
17. P. Kohli and M. P. Kumar. Energy minimization for linear envelope MRFs. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, San Francisco, California, June 2010. [1](#), [2](#), [6](#)
18. P. Kohli, L. Ladický, and P. Torr. Robust higher order potentials for enforcing label consistency. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 82(3):302–324, 2009. [2](#)
19. V. Kolmogorov and C. Rother. Minimizing non-submodular functions with graph cuts – a review. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 29(7):1274–1279, July 2007. [10](#)
20. N. Komodakis and N. Paragios. Beyond pairwise energies: Efficient optimization for higher-order MRFs. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, Miami, Florida, June 2009. [1](#)
21. F. Kschischang, B. Frey, and H.-A. Loelinger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, Feb. 2001. [1](#)
22. X. Lan, S. Roth, D. Huttenlocher, and M. Black. Efficient belief propagation with learned higher-order Markov Random Fields. In *European Conference on Computer Vision (ECCV)*, Graz, Austria, May 2006. [1](#)
23. B. Potetz and T. Lee. Efficient belief propagation for higher-order cliques using linear constraint nodes. *Computer Vision and Image Understanding*, 112(1):39–54, 2008. [2](#)
24. C. Rother, P. Kohli, W. Feng, and J. Jia. Minimizing sparse higher order energy functions of discrete variables. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, Miami, Florida, June 2009. [1](#)
25. T. Schoenemann. Probabilistic word alignment under the l_0 -norm. In *Conference Computational Natural Language Learning (CoNLL)*, Portland, Oregon, June 2011. [2](#), [7](#)
26. A. Schrijver. A combinatorial algorithm minimizing submodular functions in strongly polynomial time. *Journal of Combinatorial Theory Series B*, 80(2):346–355, Nov. 2000. [2](#)
27. D. Tarlow, I. Givoni, and R. Zemel. HOP-MAP: efficient message passing with higher order potentials. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, Sardinia, Italy, 2010. [2](#)
28. S. Vicente, V. Kolmogorov, and C. Rother. Joint optimization of segmentation and appearance models. In *IEEE International Conference on Computer Vision (ICCV)*, Kyoto, Japan, Sept. 2009. [1](#), [2](#), [4](#), [5](#), [9](#)
29. M. Wainwright, T. Jaakkola, and A. Willsky. MAP estimation via agreement on (hyper-)trees: Message-passing and linear programming approaches. *IEEE Transactions on Information Theory*, 51(11):3697–3717, 2005. [1](#)
30. T. Werner. High-arity interactions, polyhedral relaxations, and cutting plane algorithm for soft constraint optimisation (MAP-MRF). In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, Anchorage, Alaska, June 2008. [1](#), [2](#), [6](#)
31. O. Woodford, C. Rother, and V. Kolmogorov. A global perspective on MAP inference for low-level vision. In *IEEE International Conference on Computer Vision (ICCV)*, Kyoto, Japan, Sept. 2009. [1](#), [2](#), [3](#), [10](#)

32. J. Yededia, W. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. Technical report, Mitsubishi Electric Research Laboratories, Jan. 2002. [1](#)
33. J. Yededia, W. Freeman, and Y. Weiss. Constructing free energy approximations and generalized belief propagation. *IEEE Transactions on Information Theory*, 51(7):2282–2312, 2005. [1](#)
34. C. Zach, D. Gallup, J.-M. Frahm, and M. Niethammer. Fast global labeling for real-time stereo using multiple plane sweeps. In *Vision, Modeling and Visualization Workshop (VMV)*, Konstanz, Germany, Oct. 2008. [5](#)