Towards Efficient and Exact MAP-Inference for Large Scale Discrete Computer Vision Problems via Combinatorial Optimization

Jörg Hendrik Kappes[1] Markus Speth[2] Gerhard Reinelt[2] Christoph Schnörr[1]
[1]Image & Pattern Analysis Group, University of Heidelberg, Germany
[2]Discrete and Combinatorial Optimization Group, University of Heidelberg, Germany

Abstract

Discrete graphical models (also known as discrete Markov random fields) are a major conceptual tool to model the structure of optimization problems in computer vision. While in the last decade research has focused on fast approximative methods, algorithms that provide globally optimal solutions have come more into the research focus in the last years. However, large scale computer vision problems seemed to be out of reach for such methods.

In this paper we introduce a promising way to bridge this gap based on partial optimality and structural properties of the underlying problem factorization. Combining these preprocessing steps, we are able to solve grids of size 2048×2048 *in less than* 90 *seconds. On the hitherto unsolvable Chinese character dataset of Nowozin* et al. *we obtain provably optimal results in* 56% *of the instances and achieve competitive runtimes on other recent benchmark problems.*

While in the present work only generalized Potts models are considered, an extension to general graphical models seems to be feasible.

1. Introduction

We consider the problem of finding the most likely configuration of a discrete graphical model, which is equivalent to an energy minimization problem, *cf.* Eq. (1).

Since this problem is NP-hard, research has focused on approximate inference for larger problem sizes. Schlesinger [30] and Wainwright [36] proposed a linear programming (LP) relaxation based on the so-called local polytope. Several algorithms have been suggested that optimize the dual LP [30], either as a block-coordinate descent [37, 19], by subgradient or bundle methods [20, 15], or by smoothing techniques [13, 12, 28, 29]. Even if these methods solve the dual, they have to reconstruct a relaxed primal solution and then round this to a solution of the original integer program.

In another line of research, greedy move-making methods, including α-expansion [8], α-β-swap [8], and FastPD [21], have been proposed which iteratively improve an integer solution by a sequence of auxiliary max-flow problems. The efficient computation of the maximum flow has also been used to provide partial optimality on binary [27] and multi-label problems [22, 17]. This partial optimality has been employed to extend the expansion and swap methods to arbitrary functions by so-called fusion moves [23] and to reduce the problem size for the algorithms mentioned above in order to reduce their runtime [3].

In the last years combinatorial methods based on cutting-plane and branch-and-bound techniques have come more into focus in the computer vision community [4, 16, 26, 10, 34]. The main advantage of these methods is that they provide globally optimal integer solutions if no runtime restrictions are specified. The two leading methods for the MPE-task of the Probabilistic Inference Challenge 2011 [2] belong to this class.

However, combinatorial methods do not scale well and – as we will show – often do not explore the full structural and functional properties of the problems. With additional time limitations they can no longer guarantee optimality.

Contribution. We suggest a set of preprocessing methods that reduce the original problem size and can be calculated in polynomial time. One method is based on partial optimality similar to [3]. After employing partial optimality several structural properties can be used in a novel way to further reduce the problem size. Although these structural properties are simple, they are currently not fully exploited by state-of-the-art combinatorial methods. Furthermore and contrary to [3], which only uses partial optimality to speed up approximative methods, we combine reduction techniques with exact optimization methods. As a consequence, we are able to solve a significantly wider range of large scale problems to optimality that were not feasible for combinatorial methods before and demonstrate this by a comprehensive evaluation.

Organization. The paper is organized as follows: In Sec. 2 we define our problem, and in Sec. 3 we introduce several methods to solve discrete optimization problems to optimality. In Sec. 4 we will discuss methods to reduce problem sizes and to make the methods from Sec. 3 applicable for large scale problems. Finally, we demonstrate in Sec. 5 the gain of our approach for synthetic and real world problems.

1063-6919/13 $26.00 © 2013 IEEE
DOI 10.1109/CVPR.2013.229

2. Problem Formulation

Our problem is defined by a *factor graph*: a bipartite graph $G = (V, F, E)$ which consists of regular nodes V, factor nodes F and an edge set $E \subseteq V \times F$. Each node $v \in V$ is associated with a variable x_v that takes values in a discrete domain X_v. For a subset $A \subseteq V$ we denote the domain of the corresponding variables by $X_A := \bigotimes_{v \in A} X_v$. Each factor node $f \in F$ is associated with a function $\varphi_f : X_{\mathrm{nb}(f)} \to \mathbb{R}$ which is defined on the domain of the neighbors $\mathrm{nb}(f) := \{v \in V \mid (v, f) \in E\}$ of f. The cardinality of this set, $|\mathrm{nb}(f)|$, is called the *order* of the factor, factors of order 1 and 2 are called *unary* and *pairwise* factors, respectively. The order of G is the maximum order of a factor $f \in F$. A factor graph defines a *binary* problem if $|X_v| = 2$ for all $v \in V$ and a *multi-label* problem if $|X_v| > 2$ for some $v \in V$.

Using a factor graph, the energy minimization problem takes the form

$$\min_{x \in X_V} \sum_{f \in F} \varphi_f(x_{\mathrm{nb}(f)}). \tag{1}$$

Because factor graphs make the structural properties induced by the discrete Markov random field explicit, they enable efficient access to structural properties that support exact inference using combinatorial optimization.

In addition to the structural properties of the problem, properties of the functions $\varphi_f(\cdot)$ are also of major interest from the point of view of optimization, *e.g.*, submodular functions can be solved in polynomial time [25]. Problems containing unary and Potts functions only can be reformulated as a multiway cut [16] and in the case of second order binary problems as a max cut instance [11, 32]. In both cases, additional auxiliary nodes have to be introduced.

3. Exact Optimization Methods

Solving (1) is in general NP-hard. However, several authors have suggested methods that solve these problems – or special subclasses – to optimality. Since this is in general not possible in polynomial time, these methods typically do not scale, but provide fast alternatives for small and medium sized problems. We briefly introduce those that are considered in the present paper.

Integer Linear Programming (ILP) A general representation of (1) is an integer linear program. Additional to the constraints of linear programming relaxations over the local polytope, which are common in computer vision, it contains integer constraints that enforce consistency with (1). This method is implemented in [4] and imposes no further restrictions on G or $\varphi_f(\cdot)$. During optimization a sequence of linear programs is solved and integer constraints are enforced iteratively by applying cutting-plane or branch-and-bound techniques. The code of [4] is publicly available under the MIT license and uses the commercial optimization library CPLEX, which is free for academic use.

Breadth-Rotating AND/OR Branch-and-Bound (BRAOBB) Otten *et al.* suggested a depth-first search branch-and-bound algorithm over AND/OR search spaces using mini-bucket heuristics for bounding [26]. Contrary to naive depth-first search, which processes one branch of the tree after another, BRAOBB processes all branches "simultaneously" in a round-robin style. This leads to a better anytime behavior. In the Probabilistic Inference Challenge [2], BRAOBB is currently the leading method, and the source code is freely available under the GPL. As we do for the former ILP method, we do not provide any initial guess or bound to it.

Multiway Cut (MCA) Recently, Kappes *et al.* [16] suggested a transformation of generalized Potts models into a multiway cut problem and introduced a cutting-plane framework for optimization. While this method is restricted to models that contain only arbitrary first order terms and second order terms that are invariant to label permutations, it nevertheless covers many computer vision applications and provides a compact problem representation. The authors kindly provided us with the original code used in [16].

Max Cut by Branch-and-Cut (MCBC) In [5], Bonato developed a method for solving max cut problems to optimality using a branch-and-cut framework. In addition to using the standard cycle relaxation for the cut polytope he employs special separation and lifting techniques for deriving further inequalities that tighten the relaxation. The algorithm is in particular very well suited for sparse graphs and applied to computer vision problems for the first time in the present paper. The code is not publicly available, but the author kindly provided us with the possibility to run our experiments.

Max Cut using Reweighted Perfect Matching (RPM) Schraudolph *et al.* describe a method to compute a maximum cut in a planar graph by exploiting a correspondence between maximum cuts and minimum perfect matchings [32]. In [31], the algorithm is extended to non-planar graphs. In order for this to work, a so-called consistent collection of graphs that builds a cycle basis for the input graph is needed. Finding such a collection is difficult in general, however, for grid graphs the author gives one that performs well. As part of the *isinf* library, the code is free for non-commercial research and education purposes.

4. Model-Reduction

Partial Optimality (p) When dealing with computer vision problems, it is an important observation that they often contain large subproblems that can be solved efficiently. Consequently, one can calculate an optimal solution for a subset of the variables in order to reduce the problem size, as sketched in Fig. 1. This can for example be achieved by the concept of roof duality [6], as it was done for binary pairwise models [27] as well as for multi-label pairwise models [22, 17]. We found out that the MQPBO-method proposed in [17],

which is applicable to general second order problems, does not scale. Therefore we restrict ourselves to Potts models in the multi-label case and use the method proposed by Kovtun [22].

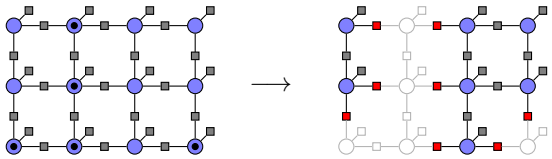

Figure 1. After applying QPBO, the nodes with known value (black dots) can be removed after modifying the factors connected to them appropriately (red).

Partial optimality has already been used in [3] to speed up *approximative* methods. By contrast, in the present paper, we systematically exploit partial optimality and problem reduction to make *exact* combinatorial methods feasible for large problem sizes. The Chinese character instances (see Sec. 5) provide a striking example – we can solve 56% of the hitherto unsolved problems.

Connected Components (c) If the factor graph G is disconnected, the individual connected components can obviously be treated separately as sketched in Fig. 2. We suggest to use a preprocessing step that detects connected components in polynomial time, *e.g.*, by using depth-first search [9], such that all of them can be solved independently.

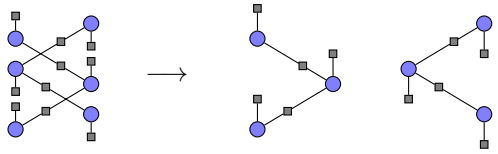

Figure 2. Treating each connected component separately can lead to huge speedups. Surprisingly, this simple observation is not taken into account by many solvers.

In Fig. 3 we show the runtimes of ILP on a problem consisting of three fully connected cliques with an increasing number of variables. ILP-c detects connected components and solves each of them independently. This leads to a huge speedup compared to ILP.

Bridge Elimination (b) We call a factor $f \in F$ a *bridge factor* if the number of connected components of G increases when f is removed. After removing a bridge factor, the resulting components can be treated separately.

When doing so while fixing the variables in $\mathrm{nb}(f)$ for all possible assignments, one side of the bridge can be shrunken to a lower order factor representing the optimal values of this subgraph, *cf.* Fig. 4. After solving the reduced problem, the optimal configuration of the shrunken part can easily be recovered.

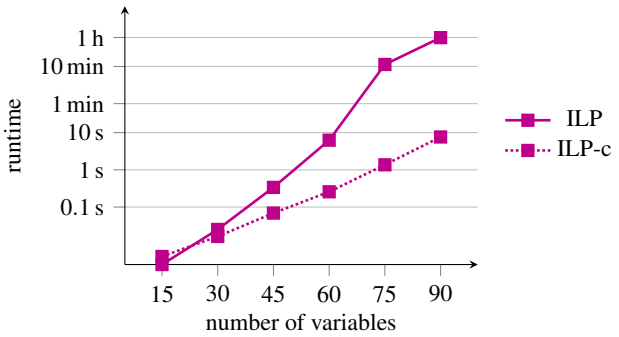

Figure 3. Average runtimes of ten binary instances per size. Each instance consists of three cliques of equal size. Standard ILP solvers are not able to capture this. Processing each connected component independently (ILP-c) leads to a significant speedup. Note that the time axis is logarithmic.

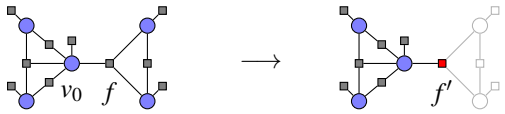

Figure 4. The right side of the original graph, including f, can be shrunken to a unary factor f' by $|X_{v_0}|$ small optimization problems. After solving the problem corresponding to the reduced graph, the full solution can be recovered.

Tentacle Elimination (t) A special case in which bridge elimination can be used very efficiently is when one side of the bridge is acyclic such that dynamic programming can be applied. Automatic detection and elimination of these acyclic substructures is done in polynomial time. We call this *tentacle elimination*, since the parts that can be eliminated are attached to the main part of the graph like tentacles, as depicted in Fig. 5. The effect on the runtime can be seen in Fig. 6.

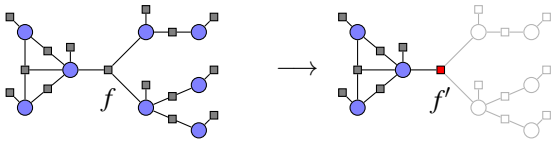

Figure 5. In cases where an acyclic subgraph (tentacles) is attached to the main part of the graph, one round of dynamic programming can be used to replace this subgraph by a single unary factor f'.

Remark. The basic idea behind bridge and tentacle elimination is also known as variable conditioning and variable elimination, *cf.* [18] and the references therein for an overview. The main difference to our work is that we suggest to use such techniques only for fast solvable substructures and not for the complete model.

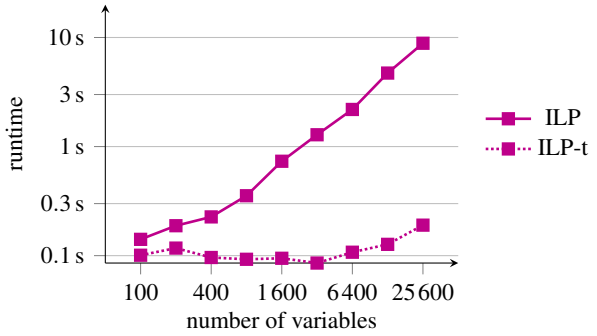

Figure 6. Average runtimes of ten binary instances per size. Each instance consists of a clique of 20 variables, all other variables are part of tentacles. Standard ILP solvers are not able to capture this. Eliminating the tentacles first (ILP-t) leads to a significant speedup. Note that both axes are logarithmic.

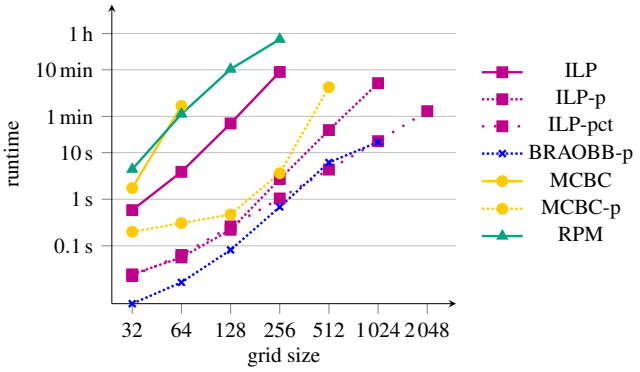

Figure 7. Average runtimes of those instances that could be solved in less than one hour (compare Fig. 8). For all methods reduction makes them applicable to larger instances. Overall, ILP-pct and BRAOBB-p perform best.

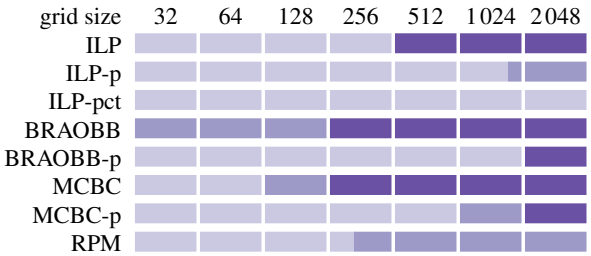

Figure 8. Fraction of the ten instances per grid size that could be solved within one hour (■), that could not be solved due to the time limit of one hour (■), and that could not be solved due to an out-of-memory error (■). Using the three proposed reduction techniques makes methods applicable to larger instances. Overall, ILP-pct scales best.

5. Evaluation and Experiments

In the following, we denote the algorithms introduced in Sec. 3 by ILP, BRAOBB, MCA, MCBC, and RPM, followed by the reduction type (p, c, t)[1] from Sec. 4 if preprocessing was applied. Additionally, we compare to TRWS [19], FastPD [21], and α-expansion [8] – all provided by the original authors of the papers. TRWS is stopped after 1 000 iterations. For evaluation we count how often a method provides the best energy value among all competing methods (*best*) and how often the gap between the energy value and the lower bound was less than 10^{-7} (*ver. opt*). For the synthetic models we use an Intel Pentium E5400, 2.7 GHz, 8 GB RAM, for the other a Xeon W3550, 3.07 GHz, 12 GB RAM. For MCBC we use a Xeon E5420, 2.5 GHz, 16 GB RAM for all experiments.

Synthetic Binary Grid Models We created problems where the underlying graph is a grid graph of size 32×32, 64×64, $\ldots$, $2\,048 \times 2\,048$. The problems are binary, *i.e.*, $|X_v| = 2$ for all $v \in V$. Unary and pairwise factors were added for all nodes and edges of the grid; the values of $\varphi_f(\cdot)$ and $\varphi_f(\cdot,\cdot)$ were drawn uniformly at random from the interval $[0,1]$ for all $f \in F$. We created ten instances per size. Problems of this type can be transformed into pure max cut problems as described in [32]. For MCBC and RPM, we transform the problems into max cut instances and solve those.

As expected, the runtimes decrease by orders of magnitudes when we apply QPBO to get partial optimal solutions first, *cf.* Fig. 7. This is not possible for RPM since it needs certain embeddings which are easy to compute for the original grid problems but not for the reduced ones. The optimality ratio was 97.6% on average.

However, we achieve another tremendous reduction of the runtime by taking into account that the reduced problems are mostly disconnected and treating the connected components as independent problems. For the $1\,024 \times 1\,024$ instances, the number of components is between 2 497 and 2 669 with sizes in the range of 4 to 83.

We do not apply the connected component reduction for BRAOBB explicitly because this is already taken into account by the method itself. Therefore, the performance of BRAOBB-p is similar to ILP-pct. For MCBC, we only apply the partial optimality reduction due to limited access.

As can also be seen in Fig. 8, the reduction methods show their full potential when applied to large scale problems. We are able to solve instances of size $2\,048 \times 2\,048$ in less than 90 seconds. Overall, BRAOBB-p and ILP-pct perform best.

Decision Tree Fields (Chinese Character Models) Decision tree fields (DTF) [24] are discriminatively learned conditional random fields (CRF). We consider the Chinese character models provided along with [24][2]. The dataset consists of 100 binary problems with first and second order terms.

[1] We do not include the general bridge elimination (b) here because we so far only implemented its special case (t).

[2] http://www.nowozin.net/sebastian/papers/DTF_CIP_instances.zip

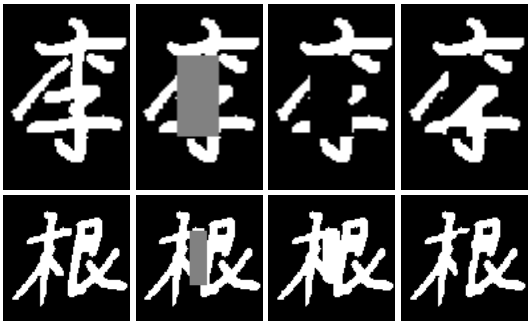

Figure 9. Examples from the Chinese character dataset. From left to right: Original image, occluded image, TRWS solution, MCBC solution.

Variables are connected to more than 30 other variables by pairwise factors. Common approximations using local polytope relaxations do not perform well on these problems, and standard ILP solvers are not able to guarantee optimality in one hour, even if partial optimality is used to reduce the problem size. For the Chinese character dataset, partial optimality has reduced the problem size from 4 992–17 856 to 502–1 093 variables.

We set a time limit of one hour per instance. Using MCBC [5] with the partial optimality reduction, we were able to verify optimality for 56 instances. Overall, we obtain superior results to all other methods, followed by ILP-pct. This is a significant progress – see Tab. 1. BRAOBB-p, which performed very good on the synthetic grid data, does not perform well on this dataset. We believe that this is caused by the larger size of the subproblems as well as the high connectivity which results in a higher treewidth.

Table 1. Results on the Chinese character dataset [24]

algorithm	avg. runtime	avg. energy	avg. bound	best	ver. opt
ILP-pct	3 581.42	−49 542.87	−50 071.87	30%	0%
BRAOBB-p	3 600.00	−49 415.55	−∞	0%	0%
MCBC-p	2 053.89	**−49 550.10**	**−49 612.38**	**92%**	**56%**
TRWS	100.13	−49 496.84	−50 119.41	2%	0%
QPBO	**0.16**	−49 501.95	−50 119.38	0%	0%
SA [24]	n/a	−49 533.02	−∞	13%	0%

Multi-Label Potts Models We also evaluate our approach on multi-label instances. To obtain partial optimality, we use the method of Kovtun [22] and therefore restrict ourselves to second order Potts functions. While more general methods exist [17, 33, 14] these do not scale as well and will be subject to further work.

First, we consider the three **color segmentation** instances used in [3]. While the standard multiway cut method [16] takes on average more than two minutes, we obtain runtimes comparable to state-of-the-art methods by using partial optimality for problem reduction and contrary to those verified globally optimal solutions, *cf.* Tab. 2. The reduction obtained by partial optimality was between 95% and 99.9%.

Table 2. Results on the three color segmentation instances from [3]

algorithm	avg. runtime	avg. energy	avg. bound	best	ver. opt
MCA	149.43	**308 472 274.3**	**308 472 274.3**	**3/3**	**3/3**
MCA-pct	**1.86**	**308 472 274.3**	**308 472 274.3**	**3/3**	**3/3**
TRWS	150.47	308 472 310.6	308 472 270.4	2/3	1/3
TRWS-pct	3.90	308 472 274.3	308 472 274.3	2/3	2/3
FastPD	**0.45**	308 472 275.0	−∞	2/3	0/3
FastPD-pct	**1.62**	308 472 274.7	−∞	2/3	0/3
α-exp	6.42	308 472 275.6	−∞	2/3	0/3
α-exp-pct	**1.72**	**308 472 274.3**	−∞	**3/3**	0/3

Figure 10. Color segmentation example from [3]. **Left:** Original image. **Middle:** Optimal segmentation. **Right:** Pixels labeled differently by TRWS (256 out of 414 720 pixels).

The differences between the energy values of the methods are quite small. Fig. 10 shows the calculated optimal labeling by MCA (middle) and the differently labeled pixels by TRWS. Differences exist in boundary regions that might be not that important for applications. However, on this dataset MCA-pct is comparable to approximative state-of-the-art methods in terms of runtime and provides optimality. Furthermore, we would like to point out that most of the runtime of MCA-pct is spent for the calculation of partial optimality.

We also apply model reduction for approximative methods as in [3]. This improves the runtimes and energies of TRWS and α-exp as can also be seen in Tab. 2. For FastPD we oberve an increase in runtime.

Second, we investigate large scale **3D MRI brain segmentation**. We use simulated 3D MRI brain data [1] and calculate the five color modes in the histogram. Our model contains five labels corresponding to the five intensity modes. The local data-term penalizes the L_1-distance between the voxel intensities and the mode intensities. We use a pairwise Potts term to penalize the boundary length using the 6-neighborhood in the 3D grid [7]. We choose a T1 pulse sequence with 3% noise relative to the brightest tissue and 20% intensity non-uniformity, *cf.* [1] for details. The simulated slice thickness was set to 3, 5, 7, and 9 mm, which results in $181 \times 217 \times 60$, 36, 26, and 20 voxel volumes, respectively.

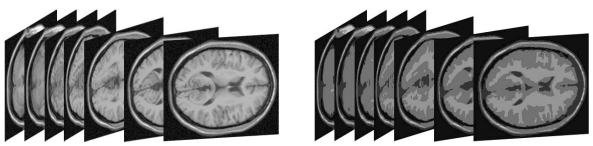

Figure 11. **Left:** Input data of the 3D MRI brain instance. It contains more than one million variables with five states. **Right:** Segmentation using the L_1-distance to the five intensity modes as data term together with a boundary length regularization.

For the instance with a thickness of 7 mm, MCA requires 20 minutes for optimization. We can reduce the runtime to less than half a minute by using partial optimality, connected components, and tentacle elimination, *cf.* Tab. 3. FastPD (both with and without reduction) and α-exp-pct provide results that are a bit faster than that of MCA-pct but worse in terms of energy. Notably, TRWS found the optimal bound but was not able to find the optimal integer solution during 1 000 iterations. For MCA-pct, 951 982 variables are eliminated by partial optimality and 7 391 by tentacle elimination. Of the remaining 2 467 subproblems, the largest one contains 45 023 variables.

Table 3. Results on the brain dataset with 7 mm slices [1]

algorithm	runtime	energy	bound	best	ver. opt
MCA	1 370.80	**12 661 506**	**12 661 506**	**yes**	**yes**
MCA-pct	21.64	**12 661 506**	**12 661 506**	**yes**	**yes**
TRWS	589.67	12 661 590	**12 661 506**	no	no
TRWS-pct	335.27	12 661 572	**12 661 506**	no	no
FastPD	**1.80**	12 663 105	$-\infty$	no	no
FastPD-pct	8.60	12 662 871	$-\infty$	no	no
α-exp	53.25	12 662 909	$-\infty$	no	no
α-exp-pct	9.91	12 662 793	$-\infty$	no	no

The runtimes for the different problem sizes are shown in Fig. 12. As already seen for the 7 mm instance, MCA-pct is quite fast and at the same time provides verified optima for all problems. MCA fails on the largest instance due to the lack of enough memory. For the approximate methods we show the gap to the optimal energy in Fig. 13. TRWS performs good but has worse runtimes than MCA-pct.

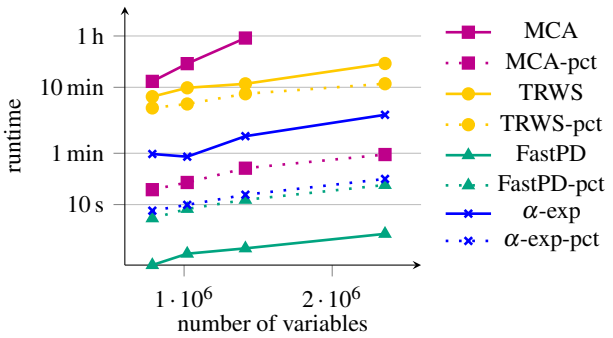

Figure 12. Runtimes for the brain dataset for different slice thicknesses which result in different numbers of variables. While for MCA, the speedup caused by model-reduction is large, it is moderate or not present for approximative methods.

6. Conclusions

We compared for the first time several combinatorial optimization methods, some not considered for computer vision problems so far, in connection with polynomial time preprocessing steps that make those methods applicable to large scale computer vision problems.

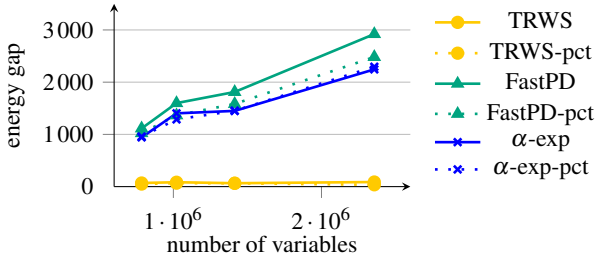

Figure 13. Energy gaps to the optimal energy for the brain dataset for different slice thicknesses which result in different numbers of variables. TRWS gives remarkably good but non-optimal solutions.

We showed that such solvers are often "blind" for simple structural properties of the problems. Using the proposed preprocessing, we were able to solve significantly larger problem instances to global optimality that have not been solved before and to reduce the runtime for exact methods substantially so as to make them applicable to real world computer vision applications.

Bridge elimination, which has not been used to far, can also be extended to bridges consisting of several factors. This might be a powerful tool for some type of models where small separators exist.

Finally, our approach requires the presence of partial optimality, and so far we only considered two methods to provide this. Recently, Swoboda *et al.* [35] published a new algorithm to obtain partial optimality in Potts models which increases the number of known variables significantly. In future work we will employ this method, consider methods for higher order factors, methods that provide stronger partial optimality criteria, as well as methods for general pairwise terms. A corresponding extension of our framework seems feasible.

Acknowledgment. We thank Thorsten Bonato, Lars Otten, and Nicol Schraudolph for discussions and their support with their respective codes. This work has been supported by the German Research Foundation (DFG) within the program "Spatio-/Temporal Graphical Models and Applications in Image Analysis", grant GRK 1653.

References

[1] BrainWeb: Simulated brain database. http://brainweb.bic.mni.mcgill.ca/brainweb/.

[2] The probabilistic inference challenge (PIC 2011). http://www.cs.huji.ac.il/project/PASCAL/.

[3] K. Alahari, P. Kohli, and P. H. S. Torr. Dynamic hybrid algorithms for MAP inference in discrete MRFs. *TPAMI*, 32(10):1846–1857, 2010.

[4] B. Andres, T. Beier, and J. H. Kappes. OpenGM: A C++ library for discrete graphical models, 2012. http://arxiv.org/abs/1206.0111.

[5] T. Bonato. *Contraction-based Separation and Lifting for Solving the Max-Cut Problem*. Optimus Verlag, 2011.

[6] E. Boros and P. L. Hammer. Pseudo-Boolean optimization. *Discrete Applied Mathematics*, 123(1–3):155–225, 2002.

[7] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *TPAMI*, 26(9):1124–1137, 2004.

[8] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *TPAMI*, 23(11):1222–1239, 2001.

[9] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press, 1990.

[10] V. Franc, S. Sonnenburg, and T. Werner. Cutting plane methods in machine learning. In *Optimization for Machine Learning*. MIT Press, 2011.

[11] P. Hammer. Some network flow problems solved with pseudo-Boolean programming. *Operations Research*, 13(3):388–399, 1965.

[12] T. Hazan and A. Shashua. Convergent message-passing algorithms for inference over general graphs with convex free energies. In *UAI*, 2008.

[13] J. K. Johnson, D. M. Malioutov, and A. S. Willsky. Lagrangian relaxation for MAP estimation in graphical models. In *Proceedings of the Annual Allerton Conference on Communication, Control, and Computing*, 2007.

[14] F. Kahl and P. Strandmark. Generalized roof duality for pseudo-Boolean optimization. In *ICCV*, 2011.

[15] J. H. Kappes, B. Savchynskyy, and C. Schnörr. A bundle approach to efficient MAP-inference by Lagrangian relaxation. In *CVPR*, 2012.

[16] J. H. Kappes, M. Speth, B. Andres, G. Reinelt, and C. Schnörr. Globally optimal image partitioning by multicuts. In *EMMCVPR*, 2011.

[17] P. Kohli, A. Shekhovtsov, C. Rother, V. Kolmogorov, and P. H. S. Torr. On partial optimality in multi-label MRFs. In *ICML*, 2008.

[18] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[19] V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *TPAMI*, 28(10):1568–1583, 2006.

[20] N. Komodakis, N. Paragios, and G. Tziritas. MRF energy minimization and beyond via dual decomposition. *TPAMI*, 33(3):531–552, 2011.

[21] N. Komodakis and G. Tziritas. Approximate labeling via graph cuts based on linear programming. *TPAMI*, 29(8):1436–1453, 2007.

[22] I. Kovtun. Partial optimal labeling search for a NP-hard subclass of (max, +) problems. In *Proceedings of the DAGM Symposium*, 2003.

[23] V. Lempitsky, C. Rother, S. Roth, and A. Blake. Fusion moves for Markov random field optimization. *TPAMI*, 32(8):1392–1405, 2010.

[24] S. Nowozin, C. Rother, S. Bagon, T. Sharp, B. Yao, and P. Kohli. Decision tree fields. In *ICCV*, 2011.

[25] J. B. Orlin. A faster strongly polynomial time algorithm for submodular function minimization. In *IPCO*, 2007.

[26] L. Otten and R. Dechter. Anytime AND/OR depth-first search for combinatorial optimization. In *SOCS*, 2011.

[27] C. Rother, V. Kolmogorov, V. Lempitsky, and M. Szummer. Optimizing binary MRFs via extended roof duality. In *CVPR*, 2007.

[28] B. Savchynskyy, S. Schmidt, J. H. Kappes, and C. Schnörr. A study of Nesterov's scheme for Lagrangian decomposition and MAP labeling. In *CVPR*, 2011.

[29] B. Savchynskyy, S. Schmidt, J. H. Kappes, and C. Schnörr. Efficient MRF energy minimization via adaptive diminishing smoothing. In *UAI*, 2012.

[30] M. I. Schlesinger. Syntactic analysis of two-dimensional visual signals in noisy conditions. *Kibernetika*, 4:113–130, 1976.

[31] N. N. Schraudolph. Polynomial-time exact inference in NP-hard binary MRFs via reweighted perfect matching. In *AISTATS*, 2010.

[32] N. N. Schraudolph and D. Kamenetsky. Efficient exact inference in planar Ising models. In *NIPS*, 2009.

[33] A. Shekhovtsov and V. Hlaváč. On partial optimality by auxiliary submodular problems. In *Control Systems and Computers*, 2011.

[34] M. Sun, M. Telaprolu, H. Lee, and S. Savarese. Efficient and exact MAP-MRF inference using branch and bound. In *AISTATS*, 2012.

[35] P. Swoboda, B. Savchynskyy, J. H. Kappes, and C. Schnörr. Partial optimality via iterative pruning for the Potts model. In *SSVM*, 2013.

[36] M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1–2):1–305, 2008.

[37] T. Werner. Revisiting the linear programming relaxation approach to Gibbs energy minimization and weighted constraint satisfaction. *TPAMI*, 32(8):1474–1488, 2010.