A Comparative Study of Modern Inference Techniques for Discrete Energy Minimization Problems

Jörg H. Kappes[1], Bjoern Andres[2], Fred A. Hamprecht[1], Christoph Schnörr[1], Sebastian Nowozin[3], Dhruv Batra[4], Sungwoong Kim[5], Bernhard X. Kausler[1], Jan Lellmann[6], Nikos Komodakis[7], Carsten Rother[3]
[1]Heidelberg University, [2]Harvard University, [3]Microsoft Research Cambridge,
[4]Virginia Tech,[5]Qualcomm Research Korea, [6]University of Cambridge, [7]Ecole des Ponts ParisTech.

Abstract

Seven years ago, Szeliski et al. published an influential study on energy minimization methods for Markov random fields (MRF). This study provided valuable insights in choosing the best optimization technique for certain classes of problems.

While these insights remain generally useful today, the phenominal success of random field models means that the kinds *of inference problems we solve have changed significantly. Specifically, the models today often include higher order interactions, flexible connectivity structures, large label-spaces of different cardinalities, or learned energy tables. To reflect these changes, we provide a modernized and enlarged study. We present an empirical comparison of 24 state-of-art techniques on a corpus of 2,300 energy minimization instances from 20 diverse computer vision applications. To ensure reproducibility, we evaluate all methods in the OpenGM2 framework and report extensive results regarding runtime and solution quality. Key insights from our study agree with the results of Szeliski et al. for the types of models they studied. However, on new and challenging types of models our findings disagree and suggest that polyhedral methods and integer programming solvers are competitive in terms of runtime and solution quality over a large range of model types.*

1. Introduction

Discrete energy minimization problems, in the form of factor graphs, or equivalently Markov or Conditional Random Field models (MRF/CRF) are a mainstay of computer vision research. Their applications are diverse and range from image denoising, segmentation, motion estimation, and stereo, to object recognition and image editing. To give researchers some guidance as to which optimization method is best suited for their MRF model, Szeliski et al. [33] conducted a comparative study on 4-connected MRF models. Along with the study, they provided a unifying software framework that facilitates a fair comparison of optimization techniques. The study was well-received in our community and has now been cited more than 600 times.

Since 2006 when the study was published, the field has made rapid progress. Modern vision problems involve more complex models, involve larger datasets and use machine learning techniques to train the model parameters and energies.

Taken together, these changes give rise to hard energy minimization problems that are fundamentally different than the ones considered by Szeliski et al. In particular, in [33] the models were restricted to 4-connected grid graphs with unary and pairwise factors only.

It is time to revisit the study of [33]. We provide a modernized comparison, updating both the problem instances and the inference techniques.

Our models are different in the following four aspects: **(1)** higher order models, e.g. factor order up to 300, **(2)** models on "regular" graphs with a denser connectivity structure, e.g. 27-pixel neighborhood, or models on "irregular" graphs with spatially non-uniform connectivity structure, **(3)** models based on superpixels with smaller number of variables, and **(4)** image partitioning models without unary terms, an unknown number of classes.

Inference methods have changed as well since 2006, often as a response to cope with more challenging models. The study [33] compared the performance of the state of the art at the time, represented by primal move-making methods, loopy belief propagation, a tree-reweighted belief propagation variant, and a set of more traditional local optimization heuristics like iterated conditional modes (ICM). We augment this set with more recently developed methods from these classes, e.g. recent move-making methods and local optimization methods. Furthermore, we use new methods which are applicable to more general, higher order models, and non-grid graph structures. In addition, we add a new class of polyhedral methods, which solve an underlying (Integer) Linear Programming formulation (LP/ILP).

Contributions We provide a modernized, follow-up study of [33] with the following aspects: **(i)** A broad collection of state-of-the-art models and inference methods. **(ii)** All models and inference techniques were wrapped into a uniform software framework, OpenGM2 [2], for reproducible benchmarking. They will be made publicly available on the project webpage[1]. **(iii)** Comprehensive and comparative evaluation of methods, along with a summary and discussion. **(iv)** We enable researchers to experiment with recent state-of-the-art inference methods on their own models.

Related Inference Studies Apart from the study [33], there are many recent articles in computer vision which compare inference techniques for a small specialized class of models, such as [4, 27, 25]. Unfortunately, the models and/or inference techniques are often not publicly available. Even if they were available, the lack of a flexible software-framework which includes these models and optimization techniques makes a fair comparison difficult. Closely related is the smaller study [8] that uses the first and now deprecated version of OpenGM. It compares several variants of message passing and move making algorithms for higher order models. In contrast to this work, polyhedral inference methods are not included and only a small number of synthetic models are used.

Outside computer vision, the Probabilistic Inference Challenge (PIC) [3] covers a broad class of models used in machine learning. We include the leading optimization techniques of PIC in our study.

Key Insights and Suggested Future Research In comparison with [33], perhaps the most important new insight is that recent, advanced polyhedral LP and ILP solvers are relevant for a wide range of problems in computer vision. For a considerable number of instances, they are able to achieve global optimality. For some problems they are even competitive in terms of overall runtime. This is true for problems with a small number of labels, variables and factors of low order that have a simple form. But even for some problems with a large number of variables or complex factor form, specialized ILP and LP solvers can be applied successfully. For problems with many variables for which the LP relaxation is not tight, polyhedral methods are not competitive. In this regime, primal move-making methods typically achieve the best results, which is consistent with the findings of [33].

Our new insights suggest two areas for future research focus. First, in order to capitalize on existing ILP solvers we could increase the use of "small" but expressive models, e.g. superpixels or coarse-to-fine approaches. Second, our findings suggest that further investigation in improving the efficient and applicability of ILP and LP solvers is warranted.

[1] http://hci.iwr.uni-heidelberg.de/opengm2/

2. Models

We assume that our discrete energy minimization problem is given in the form of a factor graph $G = (V, F, E)$, a bipartite graph, with a set of variable nodes V, a the set of all factors F, and a set $E \subset V \times F$ that defines the relation between those [23]. The variable x_a assigned to the variable node $a \in V$ lives in a discrete label-space X_a and each factor $f \in F$ has an associated function $\varphi_f : X_{ne(f)} \to \mathbb{R}$, where $\mathbf{x}_{ne(f)}$ are the variables in the neighborhood $ne(f) := \{v \in V : (v, f) \in E\}$ of the factor f, i.e. the set of variables in the *scope* of the factor. We define the order of a factor by its degree, e.g. pairwise factors have order 2, and the order of a model by the maximal degree among all factors. The energy function of the discrete labeling problem is then given as

$$J(\mathbf{x}) = \sum_{f \in F} \varphi_f(\mathbf{x}_{ne(f)}),$$

where the assignment of the variable $\mathbf{x}$ is also known as the labeling. For many applications the aim is to find a labeling with minimal energy, i.e. $\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}} J(\mathbf{x})$. This labeling is a maximum-a-posteriori (MAP) solution of a Gibbs distribution $p(\mathbf{x}) = 1/Z \exp\{-J(\mathbf{x})\}$ defined by the energy. Here, Z normalizes the distribution.

It is worth to note that we use factor graph models instead of Markov Random Field models (MRFs), also known as undirected graphical models. The reason is that factor graphs represent the structure of the underlying problem in a more precise and explicit way than MRFs can, *c.f.* [23].

2.1. Categorization of Models

One of the main attributes we use for our categorization is the *meaning* of a variable, *i.e.* if the variable is associated with a pixel, superpixel or something else. The *number* of variables is typically related to this categorization.

Another modeling aspect is the *number* of labels the variable can take. Note that the size of the label-space restricts the number of methods that are applicable, e.g. QPBO or MCBC can only be used for Boolean problems. We also classify models by *properties* of the factor-graph, *e.g.* average numbers of factors per node, mean degree of factors, or structure of the graph, *e.g.* grid structure. Finally, the *properties/type* of the functions embodied by the factors are of interest, since for some subclasses specialized optimization methods exists, *e.g.* metric energies [33] or Potts functions [19].

2.2. Benchmark Models

Table 1 gives an overview of the models summarized in this study. Note, some models have a single instance, while others have a larger set of instances which allows to derive some statistics. We now give a brief overview of all models. A detailed description of all models is available online and in the supplementary material.

	modelname	#	variables	labels	order	structure	functiontype	ref
Pixel	mrf-stereo	3	∼100000	16-60	2	grid-N4	TL1, TL2	[33]
	mrf-inpainting	2	∼50000	256	2	grid-N4	TL2	[33]
	mrf-photomontage	2	∼500000	5,7	2	grid-N4	explicit	[33]
	color-seg-N4	9	76800	3,12	2	grid-N4	potts	[29]
	inpainting-N4	2	14400	4	2	grid-N4	potts	[29]
	object-seg	5	68160	4–8	2	grid-N4	potts	[4]
	color-seg-N8	9	76800	3,12	2	grid-N8	potts	[29]
	inpainting-N8	2	14400	4	2	grid-N8	potts	[29]
	color-seg	3	21000, 424720	3,4	2	grid-N8	potts	[4]
	dtf-chinese-char	100	∼ 8000	2	2	sparse	explicit	[30]
	brain	5	400000-7000000	5	2	grid-3D-N6	potts	[1]
Superpixel	scene-decomp	715	∼ 300	8	2	sparse	explicit	[16]
	geo-surf-seg-3	300	∼ 1000	3	3	sparse	explicit	[15, 17]
	geo-surf-seg-7	300	∼ 1000	7	3	sparse	explicit	[15, 17]
	correlation-clustering	715	∼ 300	∼300	∼300	sparse	potts	[22]
	image-seg	100	500-3000	500-3000	2	sparse	potts	[6]
	3d-neuron-seg	2	7958, 101220	7958, 101220	2	sparse	potts	[9, 10]
Other	matching	4	∼ 20	∼20	2	full or sparse	explicit	[27]
	cell-tracking	1	41134	2	9	sparse	explicit	[21]

Table 1: List of datasets used in the benchmark.

Pixel-Based Models For many low-level vision problems it is desirable to make each pixel a variable in the model. For 2D images, where variables are associated to pixels in a 2D lattice, a simple form of a factor graph model connects each pixel with its four nearest neighbors using a pairwise energy. This simple form is popular and was the sole subject of the study [33]. In our study we incorporated the models *mrf-stereo*, *mrf-inpainting*, and *mrf-photomontage* from [33]. Additionally we used three models which have the same 4-connected structure, inpainting *inpainting-N4* [29], color segmentation *color-seg-N4* [29][2] and object segmentation *object-seg* [4]. All sets have a small number of labels and use Potts regularizers. In *inpainting-N4* and *color-seg-N4* this regularizer is fixed for all factors. In *object-seg*, it depends on the image-gradient. The unary terms measure the similarity to predefined class-specific color models.

From a modeling point of view such models are restricted, since they encode the assumption that each variable is conditionally independent from all others given its immediate neighbors. Hence important relations cannot be modeled by a simple grid structure. For instance, better approximations of the boundary regularization can be obtained by increasing the neighborhood [12]. Therefore, models with denser structures (both regular and irregular) as well as higher order models have been introduced in the last decade. The datasets *inpainting-N8* and *color-seg-N8* [29] include the same data-term as *inpainting-N4* and *color-seg-N4* but approximate the boundary length using an 8-neighborhood. Another dataset with an 8-neighborhood and Potts terms depending on the image-gradient is *color-seg* [4].

We also use a model with a 6-neighborhood connectivity structure in a 3D-grid. It is based on simulated 3D MRI-brain data [1], where each of the 5 labels represent color modes of the underlying histogram and boundary length regularisation [12] similar to *color-seg-N4* in 2D.

[2]The inpainting-N4/8 and *color-seg-N4/8*-models were originally used in variational approaches together with total variation regularisers [29]. A comparison with variational models is beyond the scope of this study.

(a) image (b) input (c) TRWS $J = -49768$ 100 sec. (d) MCBC $J = -49839$ 562 sec.

Figure 1: Example for a pixel based model [30].

(a) image (b) superpixels (c) MCA

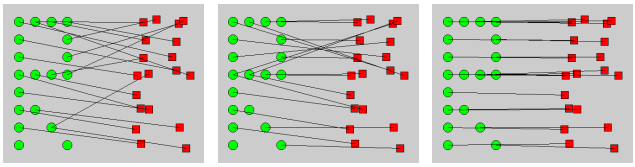
Figure 2: Example for a superpixel partition model [6]: image (left), superpixels (middle) and segmentation (right).

We also consider the task of in-painting in binary images of Chinese characters, *dtf-chinesechar* [30]. The factors are learned potentials from a decision tree field. Although each variable has only two labels, their regional sparse neighborhood structure makes the resulting inference problem challenging, *c.f.* Fig. 1.

Superpixel-Based Models In these models, all pixels that lie in the same superpixel are constrained to have the same label. This reduces the number of variables in the model and makes it attractive to add complex, higher order factors.

In the *scene-decomposition*-dataset [16] every superpixel has to be assigned to one of 8 classes. Pairwise factors between neighboring superpixels enforces likely label-neighborhoods. The datasets *geo-surf-3* and *geo-surf-7* [15, 17] are similar but have additional third-order factors, that enforce consistency of labels for three vertically neighboring superpixels.

Superpixel-Based Partition Models Beyond classical superpixel models, this study also considers a recent class of superpixel models [22, 6, 9, 10] which aim at partitioning an image without any class-specific knowledge, i.e. the corresponding energy function is invariant to permutations of the label set. Since the partition into isolated superpixels is a feasible solution, the label space of each variable is equal to the number of variables of the model, and therefore typically very large, *c.f.* Tab. 1. State-of-the-art solvers for classical models either are inapplicable or perform poorly on these models [19]. Moreover, commonly used LP-relaxations suffer from the interchangeability of the labels in the optimal solution.



(a) TRWS $J = 61.06$, 0.1 sec. (b) TRWS-LF2 $J = 34.15$, 0.5 sec. (c) ogm-ASTAR $J = 19.36$, 5.9 sec.

Figure 3: Example output for a matching model [27]: Green dots represent the variables of a fully connected graph. The discrete label assigns a green dot to a red dot (shown with a line).

The hyper-graph image segmentation dataset *correlation-clustering* [22] includes higher order terms that favor equal labels for superpixels in their scope if those are visually similar. The probabilistic image partition dataset *image-seg* [6] contains factors between pairs of superpixels. Two models for 3d neuron segmentation *3d-neuron-seg* [10] give examples for applications in large scale data.

Other Models Our benchmark includes two models in which neither pixels nor superpixels are used. The first is a non-rigid point matching problem [27], see Fig. 3. In this case the models include no unary terms, whereas the pairwise terms penalize the geometric distortion between pairs of points. The second model is a cell tracking model [21]. Variables correspond to tracks of cells in a video sequence. Since a track can either be active or dormant, the variables are binary. Higher order factors are used to model the likelihood of a "splitting" and "dieing" event of a cell.

3. Inference Methods

We consider more than 24 different inference methods for evaluation. The selection of methods is representative of the state of the art in the field. We now give a brief overview of these methods. As for the model instances, we provide a more detailed description along with parameter settings in the supplementary material.

Polyhedral and Combinatorial Methods A large class of algorithms solves a linear programming relaxation (LP) of the discrete energy minimization problem. Perhaps the most commonly used relaxation is the LP relaxation over the *local polytope*. For small instances this can be done by standard LP-solvers *e.g.* ogm-LP-LP [5]. For large problems this is no longer possible and special solvers have been proposed that optimize a dual formulation of the problem. A famous example is the block-coordinate-ascent method TRWS [24], which, however, can get stuck in local fix points. In contrast, subgradient methods (ogm-SUBGRAD-A) [18] and bundle methods [18] with adaptive (ogm-BUNDLE-A) or heuristic (ogm-BUNDLE-H) stepsize are guaranteed to converge to the optimum of the relaxed dual[3]. In both cases primal integer solutions are reconstructed from the subgradients. Related to polyhedral methods are Integer Linear Programs (ILPs). These include additional integer constraints and guarantee global optimality, contrary to the methods based on LP-relaxations. Solutions of ILPs are found by solving a sequence of LPs and either adding additional constraints to the polytope (cutting plane techniques), or branching the polytope into several polytopes (branch-and-bound techniques). We evaluate three state-of-the-art ILP solvers: IBM CPLEX wrapped by OpenGM2 [5] (ogm-ILP), the current best performing method in the PIC called breath-rotating and/or branch-and-bound [31] (BRAOBB), and the AStar-Method (ogm-ASTAR) [11]. To reduce the large memory requirements which come along with the vectorization of the objective for the LP, we also consider the multicut-representation introduced by Kappes *et al*. [19]. This multicut solver (MCA) can only be applied for functions which includes terms that are either invariant under label permutations or of first-order. Additionally, we tested a variant that only add facet-defining constraints during the cutting plane procedure, called MCA-fdo. We also consider a relaxed multicut version [22] (MCR) only applicable to partition problems, and a max-cut solver (MCBC) for pairwise binary problems.[4]

Message Passing Methods Message passing methods are simple to implement and can be easily parallelized, making them a popular choice in practise. Polyhedral methods can often be reformulated as a message passing method, *e.g.* TRWS [24]. Also its non-sequential version TRBP [34] can be written as a message passing algorithm. TRBP can be applied to higher order models but has no convergence guarantees. Practically it works well if sufficient message damping [34] is used. Maybe the most popular message passing algorithm is loopy belief propagation (LBP). While LBP converges to the global optima for acyclic models, it is only a heuristic for general graphs, which turns out to perform reasonably well in practise. We evaluate the parallel (LBP) and sequential (BPS) versions from [33], as well the general higher order implementation using parallel updates (ogm-LBP) from [5]. For parallel methods we use message damping.

Max-Flow and Move-Making Methods In some cases network flow algorithms exist that are equivalent to LP-relaxations. These converge in finite time and are usually much faster. For submodular second-order binary models min s-t cut is such an example. For general second-order binary models, QPBO [32] solves the LP relaxation over the local polytope and provides a persistence certificate. An-

[3] Here we consider spanning trees as subproblems such that the relaxation is equivalent to the local polytope relaxation.

[4] The latter two methods are not publicly available and results were kindly provided by the authors.

other class of common methods applies a greedy minimization over the label space by solving a sequence of max-flow problems. Three members of this class are α-Expansion (EXPANSION), α-β-Swap (SWAP) [13, 26] and FastPD [28]. We also consider a generalization of α-Expansion for general problems and higher order factors known as α-Fusion (FUSION), which fusion-moves and the order-reduction of Fix *et al*. [14]. The Lazy Flipper [7] algorithm is similar in that it iteratively applies a greedy search over local subsets. It converges to a configuration which is optimal within a Hamming distance in label space, without using max-flow methods. The Lazy Flipper relies heavily on initialization, for which we use standard methods.

4. Experimental Setup

Our benchmark builds on the success of the OpenGM2 framework [2, 5], which provides custom implementations and wrappers to existing implementations, for several state-of-art techniques. In the following, we use prefixes in method names to indicate the source of the implementation. Prefixes *ogm* and *mrf* are used to denote methods whose source implementation is [5] and [33] respectively. Lack of a prefix in some method names indicates that we used available code from the corresponding authors and wrote OpenGM2 wrappers. This is always possible because in contrast to other graphical-model libraries, OpenGM2 imposes no restrictions on the class of allowed functions or topology of the models. Furthermore, the use of a common platform provides a uniform representation of models, as well as a file format that is sufficiently general to store all models in this study with affordable file sizes. All models used in this study will be made available in the OpenGM2 file format. All experiments were conducted on a Xeon W3550 machine with 3.07GHz CPU and 12 GB RAM. In an attempt to perform a fair comparison, each solver was allowed to use only one thread, and only 3 of the 4 cores were used. The evaluation of the parallelizability of the methods is beyond the scope of this comparison. Furthermore, in order to keep computational resources in check, we stop bundle, subgradient and message passing methods after 1000 iterations and other polyhedral and combinatorial methods, which do not have an applicable stopping criteria based on iteration, after 1 hour, if they have not yet converged.

Some evaluated algorithms do not use the OpenGM2 data structures and rather convert the instance into their own internal data structures. In order to evaluate runtime, we do not measure the time taken for an initial setup of memory buffers. During optimization we store the value of the current integer solution and the lower bound after each iteration, if applicable. The runtime of the algorithm does not include the overhead for computing and maintaining such intermediate statistics. Consequently, we can compare upper and lower bounds of the algorithms at various runtimes. It is important to note that comparing algorithms solely in terms of runtime has the effect of favoring implementation efficiency over algorithmic efficiency. For instance, a general implementation of the same algorithm will typically be significantly slower than a specialized implementation that makes assumptions about the problem structure (*e.g*. only pairwise terms). In fact, our benchmark does include highly optimized implementations for special problem-subclasses, *e.g*. those from [33], as well as general, less optimized research code. We observe a speed-up factor of $10 - 100$ for an optimized implementation compared to a general implementation.

Clearly, not all methods can be used for all types of models. We made our best effort to apply methods to as many models as possible. The supplementary material provides detailed information about the data structures used by each algorithm, specific restrictions on problem-classes and other considerations.

5. Evaluation

Due to lack of space, we only provide a brief summary of the benchmark results here. Detailed results for all instances, including plots of energy values and bounds versus runtime, are provided in the supplemental material and will be made publicly available on the project webpage[5].

It is important to note that in contrast to [33], our comparison includes techniques with significantly different internal data structures and underlying motivations. Specifically, some solvers are specialized implementations for a certain class of problems (*e.g*. grids with Potts functions), while others make no assumptions about the problem and tackle the general case (*i.e*. arbitrary functions and order). While both paradigms have merit, it becomes challenging to quantify their relative performance in a fair manner. Due to the algorithmic complexity we expect a speedup of ~ 100 for specialized implementations of methods with prefix *ogm*, all other methods should be comparable.

Over all our experiments, we found that FastPD is typically the fastest method, on problems where it is applicable. Even though it only provides approximate solutions, they are often of high quality. For difficult models like *dtf-chinesechar* and *matching*, the exact solution is significantly better than approximate ones, both in terms of energy and quality. For some models, surprisingly, the exact methods are *faster* than currently reported state-of-the-art methods. Solutions from these exact methods are used for evaluating the sub-optimality of the approximate solutions.

We also mention that even when they are not the fastest, message passing, polyhedral, and exact polyhedral methods often provide very good results quickly, and then spend most of the time to verify optimality or achieve a stopping criterion.

[5] http://hci.iwr.uni-heidelberg.de/opengm2/

The tables below give a snapshot of the best performing methods for various models. We report runtime (**mean runtime**), objective value achieved by the final integer solution (**mean value**), and lower bound achieved (**mean bound**). All three quantities are averaged over all instances for a model. Furthermore, we report how often (in terms of percentage) an algorithm returned the best[6] (not necessary optimal) integer solution among all results (**best**) and how often the method was able to "verify optimality" by achieving a relative integrality gap smaller than 10^{-8} (**ver. opt**). Note that for some instances, no method was able to verify optimality.

Second-Order Models Table 2 summarizes the results for the stereo labeling problems from [33]. For instances of this problem, methods like ogm-BUNDLE-A that solve the LP-relaxation perform best. The mrf-TRWS method from [33] produces the best solution on average, but does not solve the LP-relaxation optimally, since the average lower bound is not the tightest. Applying Lazy Flipper on top (TRWS-LF2) improves the solution further.

Table 2: mrf-stereo (3 instances)

algorithm	mean run time	mean value	mean bound	best	ver. opt
FastPD	**4.47 sec**	1614255.00	$-\infty$	0.00	0.00
FastPD-LF2	296.38 sec	1611484.33	$-\infty$	0.00	0.00
mrf-EXPANSION	13.04 sec	1614353.00	$-\infty$	**33.33**	0.00
mrf-TRWS	296.22 sec	1587928.67	1584746.53	0.00	0.00
ogm-BUNDLE-A	8458.49 sec	1610985.00	1584776.15	**33.33**	**33.33**
ogm-SUBGRAD-A	8533.41 sec	1752958.00	1578421.00	**33.33**	**33.33**
TRWS-LF2	668.47 sec	**1587040.00**	1584746.53	**33.33**	0.00

In contrast to the previous case, primal move-making methods outperform LP-relaxation methods for the *mrf-photomontage* model, also from [33]. Table 3 provides the details. We believe this is because some unary terms have infinite values. As observed in [33], mrf-EXPANSION is both the fastest and achieves the best results on average. The same behavior can be seen for the synthetic inpainting instances *inpainting-N4/8*. The instance "inverse" is constructed such that the LP-relaxation over the local polytope is not tight. In this case pure primal move-making methods are superior, since they do not face the rounding problem (see supplementary material for details).

Table 3: mrf-photomontage (2 instances)

algorithm	mean run time	mean value	mean bound	best	ver. opt
mrf-EXPANSION	**8.54 sec**	**168220.00**	$-\infty$	**100.00**	0.00
mrf-SWAP	**11.35 sec**	**180345.00**	$-\infty$	0.00	0.00
mrf-TRWS	253.20 sec	1243144.00	166827.07	0.00	0.00
ogm-BUNDLE-H	4771.35 sec	599206.00	111100.35	0.00	0.00
ogm-SUBGRAD-A	4734.20 sec	3846787.00	26005.24	0.00	0.00
TRWS-LF2	431.34 sec	735193.00	166827.12	0.00	0.00

[6]For numerical reasons, a solution is considered to be "best" if its value deviates by less than 10^{-8} from the best value.

In Table 4, we analyze the color-seg-4 model that has fewer variables and a simpler Potts regularization. In this case, we are able to calculate the globally optimal solution using MCA. This is also true for other models with similar characteristics, *e.g. color-seg*, *object-seg*, *color-seg-n8*, and *brain*. Even the complex *pfau*-instance could be solved to optimality in 3 hours. In this case, LP-based methods are superior in terms of objective values, but EXPANSION, SWAP and FastPD converged to somewhat worse but reasonable solutions very quickly.

Table 4: color-seg-n4 (9 instances)

algorithm	mean run time	mean value	mean bound	best	ver. opt
FastPD	**0.35 sec**	20034.80	$-\infty$	0.00	0.00
FastPD-LF2	13.61 sec	20033.21	$-\infty$	0.00	0.00
mrf-EXPANSION	**1.24 sec**	20031.81	$-\infty$	0.00	0.00
mrf-SWAP	**0.86 sec**	20049.90	$-\infty$	0.00	0.00
mrf-TRWS	33.15 sec	**20012.18**	**20012.14**	88.89	77.78
ogm-BUNDLE-A	692.39 sec	20024.78	20012.01	77.78	77.78
ogm-BUNDLE-H	1212.24 sec	20012.44	**20012.13**	77.78	22.22
ogm-SUBGRAD-A	1179.62 sec	20027.98	20011.57	66.67	11.11
MCA	982.36 sec	20527.37	19973.25	88.89	88.89
MCA-6h	1244.30 sec	**20012.14**	**20012.14**	**100.00**	**100.00**

All models so far consisted of truncated convex pairwise terms. Arbitrary pairwise terms can lead to optimization problems that are significantly harder to solve, as we found in *dtf-chinesechar* in Table 5. In this case, the pairwise terms are learned and happen to be a mix of attractive and repulsive terms. Although these are medium sized binary problems, the relaxations over the local polytope are no longer as tight. Only the advanced polyhedral method (MCBC) [20] was able to solve some (56) instances to optimality.

Table 5: dtf-chinesechar (100 instances)

algorithm	mean run time	mean value	mean bound	best	ver. opt
BPS	72.85 sec	-49537.08	$-\infty$	19.00	0.00
MCBC	2053.89 sec	**-49550.10**	**-49612.38**	**91.00**	**56.00**
ogm-ILP	3580.93 sec	-49536.59	-50106.17	8.00	0.00
QPBO	**0.16 sec**	-49501.95	-50119.38	0.00	0.00
SA [30]	n/a	-49533.02	$-\infty$	13.00	0.00
TRWS	100.13 sec	-49496.84	-50119.41	2.00	0.00
TRWS-LF2	106.94 sec	-49519.44	-50119.41	11.00	0.00

The matching problems in Table 6 have very few variables, which is ideal for sophisticated ILP solvers. Indeed, we observe that pure branch-and-bound algorithms like BRAOBB or ogm-ASTAR can achieve global optimality relatively quickly. Again, standard LP-solvers do not perform well, since the relaxation is not very tight. Lazy flipping, as a post-processing step, can help significantly in these situations, *c.f.* Fig. 3. Fusion moves with α-proposals does not work well for matching instances. Generating problem-specific proposals might overcome this problem.

Table 6: matching (4 instances)

algorithm	mean run time	mean value	mean bound	best	ver. opt
BPS	**0.17 sec**	40.26	$-\infty$	25.00	0.00
BRAOBB	**3.50 sec**	**21.22**	**21.22**	**100.00**	**100.00**
FUSION	**0.02 sec**	1465000000000.00	$-\infty$	0.00	0.00
ogm-ASTAR	8.78 sec	**21.22**	**21.22**	**100.00**	**100.00**
ogm-BUNDLE-H	5.58 sec	7500000055.96	15.91	0.00	0.00
ogm-ILP	1287.07 sec	23.04	20.44	75.00	75.00
ogm-ILP-6h	1694.34 sec	**21.22**	**21.22**	**100.00**	**100.00**
ogm-LP-LP	33.17 sec	102500000036.76	16.35	0.00	0.00
TRWS	**0.17 sec**	64.29	15.22	0.00	0.00
TRWS-LF2	**0.76 sec**	32.38	15.22	0.00	0.00

Second-Order Models - Superpixel Models based on superpixels differ from pixel-based models in two important ways. First, they have a small number of variables and second, they often have very strong unary data-terms that make local polytope relaxations often tight. Consequently, ILP-solvers become not only feasible, they outperform state-of-the-art methods in terms of runtime. One example of this behavior is the scene-decomposition model where ogm-ILP performs best. Table 7 shows the results.

Table 7: scene-decomposition (715 instances)

algorithm	mean run time	mean value	mean bound	best	ver. opt
BPS	0.17 sec	-866.73	$-\infty$	79.16	0.00
BRAOBB	28.31 sec	**-866.93**	$-\infty$	**100.00**	99.86
FUSION	**0.07 sec**	-866.85	$-\infty$	82.10	0.00
ogm-BUNDLE-H	0.91 sec	**-866.93**	**-866.93**	**100.00**	94.13
ogm-ILP	**0.11 sec**	**-866.93**	**-866.93**	**100.00**	**100.00**
ogm-LP-LP	**0.09 sec**	-866.92	**-866.93**	99.58	99.58
TRWS	0.17 sec	-866.92	**-866.93**	99.58	99.58

Higher Order Models Higher order superpixel models exhibit similar behavior as the second-order superpixel models. As long as the number of labels and the order of factors is small, the linear objective is small and ILP-solvers outperform alternative methods. Table 8 shows an example, where again ogm-ILP is the best and only FUSION converge faster, but not always to optimal solutions.

Table 8: geo-surf-7 (300 instances)

algorithm	mean run time	mean value	mean bound	best	ver. opt
BRAOBB	1031.34 sec	478.96	$-\infty$	82.67	74.33
FUSION	**0.28 sec**	477.83	$-\infty$	85.67	0.00
ogm-BUNDLE-H	97.91 sec	**476.95**	476.86	99.67	60.00
ogm-SUBGRAD-A	129.24 sec	479.26	473.16	53.33	4.00
ogm-ILP	**0.87 sec**	**476.95**	**476.95**	**100.00**	**100.00**
ogm-LBP	3.51 sec	498.45	$-\infty$	22.00	0.00
ogm-LP-LP	2.27 sec	**476.95**	476.94	99.67	99.67

For the cell tracking instance, shown in Table 9, ILP-methods clearly outperform all the alternatives. Only the LP-solver manages to find a solution that satisfies the constraints, included as soft-constraint in the model. Applying the Lazy Flipper as a post-processing can overcome this problem, as shown for LBP.

Table 9: cell-tracking (1 instances)

algorithm	mean run time	mean value	mean bound	best	ver. opt
ogm-LBP-LF2	308.83 sec	7515575.61	$-\infty$	0.00	0.00
FUSION	11.12 sec	34335812.82	$-\infty$	0.00	0.00
ogm-BUNDLE-H	1068.11 sec	107553778.57	7501875.98	0.00	0.00
ogm-ILP	32.77 sec	**7514421.21**	**7514421.21**	**100.00**	**100.00**
ogm-LBP	30.47 sec	407520058.41	$-\infty$	0.00	0.00
ogm-LP-LP	**3.26 sec**	7516359.61	7513851.52	0.00	0.00

Partition Models Partition models, shown in [19], are challenging for classical MRF techniques, because any label permutation of an optimal solution is also an optimal solution. An efficient way to solve these problems is by transforming them into a multicut problem. We apply the solver suggested in [19, 6] and also apply a variant that only considers facet defining constraints (-fdo). For the correlation-clustering dataset, which contains terms of order up to 300, we also report the original results obtained by an outer relaxation (MCR) [22]. Table 10 shows that the approximate method is a factor of 3-times faster than exact methods but can verify optimality only in 10% of the cases. We observe for all partition models using only the facet-defining constraints yields better runtimes. Overall, we have optimal solutions for 3 different structured partition models, *c.f.* Tab. 1.

Table 10: correlation-clustering (715 instances)

algorithm	mean run time	mean value	mean bound	best	ver. opt
MCR	**0.38 sec**	-624.35	-629.03	16.36	10.21
MCA	1.14 sec	**-628.16**	**-628.16**	**100.00**	**100.00**
MCA-fdo	1.04 sec	**-628.16**	**-628.16**	**100.00**	**100.00**

6. Conclusions

We presented a large set of discrete energy minimization problems for computer vision applications whose variety reflects progress of the field concerning both modelling and MRF-based inference during the last decade.

For small and moderate problem sizes, advanced integer programming methods using cutting-plane and branch-and-bound techniques not only provide a global optimality certificate, but also tend to outperform alternative approximative methods in terms of speed.

This is not the case yet for large-scale problems. In such cases, whenever move making methods like α-expansion or FastPD can be applied, they often efficiently provide solutions that are accurate enough for many applications. Otherwise, dedicated methods based on LP-relaxation provide lower bounds on minimal energies and in turn reasonable integer-valued solutions. Finally, for sophisticated models LP-relaxations may not be tight enough. In these cases, pure primal move making methods are faster and return better solutions.

This work also includes a unifom and flexible software framework that offers a broad range of inference algorithms to researchers in computer vision. The project website provides all models and methods considered and gives the opportunity to add models and results of other inference methods online. We hope this will encourage researchers to construct more complex models and in turn novel optimization methods beyond second-order metric grid models.

Acknowledgment We thank Rick Szeliski and Pushmeet Kohli for inspiring discussions. 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] OpenGM2. http://hci.iwr.uni-heidelberg.de/opengm2/.

[3] The probabilistic inference challenge (PIC2011). http://www.cs.huji.ac.il/project/PASCAL/.

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

[5] B. Andres, T. Beier, and J. H. Kappes. OpenGM: A C++ library for discrete graphical models. *ArXiv e-prints*, 2012.

[6] B. Andres, J. H. Kappes, T. Beier, U. Köthe, and F. A. Hamprecht. Probabilistic image segmentation with closedness constraints. In *ICCV*, 2011.

[7] B. Andres, J. H. Kappes, T. Beier, U. Köthe, and F. A. Hamprecht. The lazy flipper: Efficient depth-limited exhaustive search in discrete graphical models. In *ECCV*, 2012.

[8] B. Andres, J. H. Kappes, U. Köthe, C. Schnörr, and F. A. Hamprecht. An empirical comparison of inference algorithms for graphical models with higher order factors using OpenGM. In *DAGM*, 2010.

[9] B. Andres, U. Köthe, T. Kroeger, M. Helmstaedter, K. L. Briggman, W. Denk, and F. A. Hamprecht. 3D segmentation of SBFSEM images of neuropil by a graphical model over supervoxel boundaries. *Medical Image Analysis*, 16(4):796–805, 2012.

[10] B. Andres, T. Kröger, K. L. Briggman, W. Denk, N. Korogod, G. Knott, U. Köthe, and F. A. Hamprecht. Globally optimal closed-surface segmentation for connectomics. In *ECCV*, 2012.

[11] M. Bergtholdt, J. H. Kappes, S. Schmidt, and C. Schnörr. A study of parts-based object class detection using complete graphs. *IJCV*, 87(1-2):93–117, 2010.

[12] Y. Boykov. Computing geodesics and minimal surfaces via graph cuts. In *ICCV*, 2003.

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

[14] A. Fix, A. Gruber, E. Boros, and R. Zabih. A graph cut algorithm for higher-order Markov random fields. In *ICCV*, 2011.

[15] A. C. Gallagher, D. Batra, and D. Parikh. Inference for order reduction in Markov random fields. In *CVPR*, 2011.

[16] S. Gould, R. Fulton, and D. Koller. Decomposing a scene into geometric and semantically consistent regions. In *ICCV*, 2009.

[17] D. Hoiem, A. A. Efros, and M. Hebert. Recovering occlusion boundaries from an image. *IJCV*, 91(3):328–346, 2011.

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

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

[20] J. H. Kappes, M. Speth, G. Reinelt, and C. Schnörr. Towards efficient and exact MAP-inference for large scale discrete computer vision problems via combinatorial optimization. In *CVPR*, 2013.

[21] B. X. Kausler, M. Schiegg, B. Andres, M. Lindner, H. Leitte, L. Hufnagel, U. Koethe, and F. A. Hamprecht. A discrete chain graph model for 3d+t cell tracking with high misdetection robustness. In *ECCV*, 2012.

[22] S. Kim, S. Nowozin, P. Kohli, and C. D. Yoo. Higher-order correlation clustering for image segmentation. In *NIPS*. 2011.

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

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

[25] V. Kolmogorov and C. Rother. Comparison of energy minimization algorithms for highly connected graphs. In *ECCV*, pages 1–15, 2006.

[26] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? In *ECCV*, 2002.

[27] N. Komodakis and N. Paragios. Beyond loose LP-relaxations: Optimizing MRFs by repairing cycles. In *ECCV*, 2008.

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

[29] J. Lellmann and C. Schnörr. Continuous Multiclass Labeling Approaches and Algorithms. *SIAM J. Imag. Sci.*, 4(4):1049–1096, 2011.

[30] S. Nowozin, C. Rother, S. Bagon, T. Sharp, B. Yao, and P. Kohli. Decision tree fields. In *ICCV*, pages 1668–1675. IEEE, 2011.

[31] L. Otten and R. Dechter. Anytime AND/OR depth-first search for combinatorial optimization. In *Proceedings of the Annual Symposium on Combinatorial Search (SOCS)*, 2011.

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

[33] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. A comparative study of energy minimization methods for Markov random fields with smoothness-based priors. *IEEE PAMI*, 30(6):1068–1080, 2008.

[34] M. J. Wainwright, T. Jaakkola, and A. S. Willsky. MAP estimation via agreement on trees: message-passing and linear programming. *IEEE Trans. Inf. Theory*, 51(11):3697–3717, 2005.